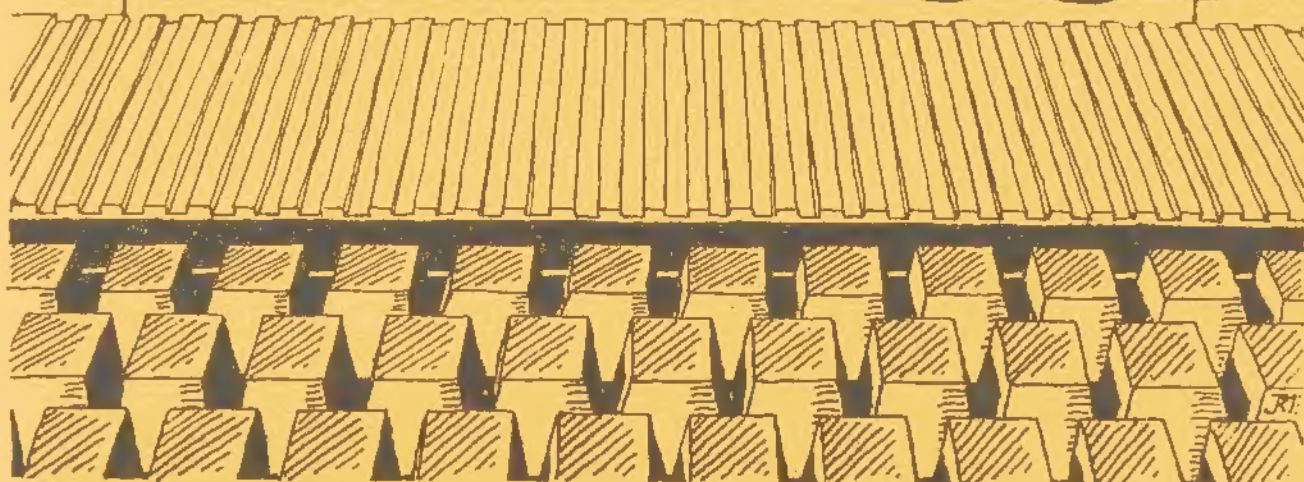
# ACORN NIEUWS

JAARGANG
NUMMER



**Bestuur:**

Voorzitter:
K. Ottes
Wilkemaheerd 67
9736 BN Groningen
Telf. 050-417042

Secretaris:
N. Stad
Plataanweg 47.
1544 PB Zaandijk.
Telf. 075-280808

Penningmeester:
Th. van Kempen.
Het Puyven 71.
5672 RB Nuenen.
Telf. 040 - 836210

Clubwinkel:
N. Stad
Plataanweg 47
1544 PB Zaandijk
Telf. 075 - 280808

Hard-ware cie.:
P. Ehrlich
Roostenlaan 266
5644 BS Eindhoven.
Telf. 040 - 114183

Redaktie Acorn Nieuws.:
H. Reinders.
Leeuwarderstraat 8.
9718 HX Groningen.
Telf. 050 - 125458

Giro 5244293 BANK 52.84.69.010
Beide t.n.v. Penningmeester Acorn computerclub.
Nuenen.

Redactie ACORN NIEUWS:
Rudi van Drunen - hardware
Hans Marks - software
Henk Reinders - layout

Redactie Acorn Nieuws:
Postbus 1050
9700 BA Groningen.
Telf 050 - 125458

Bandjesarchief:
P. Grevelt.
Swalmstraat 12
1784 CP Den Helder
Telf 02230 - 37060

Datasheets:
G. Akkermans.
Wikke 1
1273 BR Huizen.
Telf 02152 - 60294.

Drukwerkarchief:
F. Monsato
Biesbosch 153
8032 VD Zwolle.

Uiterste datum inlevering copy:

NR 2. 25 - 1 - 1985.
NR 3. 8 - 3 - 1985.
nr 4. 19 - 4 - 1985.
NR 5. 5 - 7 - 1985.
NR 6. 16 - 8 - 1985.
NR 7. 27 - 9 - 1985.
NR 8. 8 - 11 - 1985.

DE CLUB-WINKEL:

| | |
|---|---|
| Geheugenkaart: 16 kByte extra in de ATOM | fl. 40,00 |
| Schakelkaart: meerdere EPROM's op Axxx (incl.74LS133) | fl. 50,00 |
| Programmerkaart: zelf programmeren van EPROMs | fl. 21,00 |
| Big Benny print:incl. Clock IC MSM 5832RS | fl. 42.50 |
| Los IC MSM 5832RS | fl. 32.50 |
| Minischakelkaart | fl. 16.00 |
| Herdruk ACORN NIEUWS 1982: 97 pag. wetenswaardigheden | fl. 6.00 |
| Jaargang 1983: totaal ruim 450 pag. | fl. 30.00 |
| Jaargang 1984: | fl. 35.00 |
| ATOM-WARE deel 1: machinetaal op de ATOM 98 pag. | fl. 6.00 |

**Bestellen:**
Bij Uw regionale penning-meester.
Rechtstreeks bij de federatieve penningmeester, dan fl. 4.00 extra voor porto-kosten.

ACORN NIEUWS is een uigave van de fed. acorn computerclubs n/b
verschijnt 6 - 8 keer per jaar.

Zoals in ACORN NIEUWS al te lezen was is het bestuur flink gewijzigd en uitgebreid. Deze uitbreiding heeft ook een herziening van de taakverdeling tot gevolg gehad. We kunnen ons voorstellen dat er leden zijn die niet precies meer weten bij wie ze zich moeten vervoegen als ze met een bepaald punt zitten. Daarom hierbij een klein overzicht van de hanteren spelregels, zodat niet de verkeerde personen worden benaderd.

1) Adreswijzigingen en opzegging lidmaatschap.
Licht uw eigen leden administrateur in en schrijf een briefje of een verhuiskaart aan de federatieve leden administratie. Deze wordt beheerd door:
K. Ottes
Wilkemakeerd 67
9736 BN GRONINGEN

2) Bestelling clubartikelen.
Doe dit via uw eigen penningmeester of met bijstorting van f.4.00 met vermelding van de gewenste artikelen bij de federatieve penningmeester. U krijgt dan de artikeln zo spoedig mogelijk franko thuisgestuurd.
Deze bestelling kan geschieden via:
Giro 5244294 of Bank 52.84.69.010
Beide t.n.v. Penningmeester ACORN computerclub.
EINDHOVEN.

3) Klachten over clubartikelen en niet ontvangen nummers ACORN NIEUWS:
N.STAD
Plataanweg 47
1544 PB ZAANDIJK.
075-280808 ('savond na 19.00 uur)

4) Reaktie's over en COPY voor ACORN NIEUWS:
Deze het liefst op schijf of casette en voorzien van de naam van de auteur aan:
Redaktie ACORN NIEUWS
Postbus 1050
9700 BA GRONINGEN.

5) Vragen en suggestie's betreffende hardware zaken.
Deze kunnen, vragen alleen als ze niet in de regio opgelost kunnen worden, schriftelijk gericht worden aan:
P. Ehrlich
Roostenlaan 266
5644 BS EINDHOVEN

6) Andere federatieve zaken.
Bestuurswijzigingen in de regio en zaken betreffende de federatie kunnen schriftelijk gericht worden aan:
Secretariaat F.A.C.
Plataanweg 47
1544 PB ZAANDIJK.

7) Alle andere zaken.
Alle zaken die niet hierboven zijn genoemd, zoals archieven, dienen in de regio afgehandeld te worden.

Als iedereen zich aan bovengenoemde spelregels houdt, dan kan er optimaal gewerkt worden en worden vertragingen voorkomen.
Dan zijn er nog twee punten die mij van het hart moeten.
Nieuwe leden kunnen bij de regionale penningmeester opgegeven worde, deze schrijft dan een acceptgirokaart uit met:

a. naam, adres, postcode en woonplaats.
b. de regio.
c. het jaar waarvoor de contributie wordt betaald.

Of een nieuw lid schrijft een overschrijvingskaart uit op boven genoemde giro- of bankrekening, met daarop:

a. naam, adres, postcode en woonplaats.
b. de regio
c. het jaar waarvoor de contributie wordt betaald.

We dringen er op aan om al deze gegevens te vermelden, anders moet de penningmeester er achter aan om deze gegevens te achterhalen, en dat kost vrij veel tijd.
Dan als tweede, als een aantal mensen met een project bezig gaan, b.v. een nieuwe disk controller kaart ontwikkelen, schrijf dan even een briefje aan de hardware commissie, dan zijn zij op de hoogte en kunnen dit soort ontwikkelingen enigszins coordineren. Houdt ze ook regelmatig op de hoogte van de ontwikkelingen.

Met ingang van dit nummer wordt de adressering van ACORN NIEUWS door onszelf gedaan. Daartoe is een programma ontwikkeld, waarbij ieder lid voorzien is van een lidnummer. De twee laatste letters geven de regio aan en de betekenis is als volgt:

| | | |
|---|---|---|
| AH = Arnhem | BO = Brabant | CT = Centrum |
| BE = Belgie | DE = Delft | DH = Den Haag |
| LB = Limburg | NH = Noord Holland | NO = Noord |
| OG = Overijsel/Geld. | RD = Rotterdam | TW = Twente |
| ZL = Zeeland | | |

Indien de code op de adreswikkel of Uw adres onjuist is, geef dit dan door aan de leden administrateur van de federatie.
Indien U dit voor 15 februari 1985 doet dan wordt de afdracht met de regio alsnog gecorrigeerd.

## ALGEMENE LEDENVERGADERING VAN DE REGIO NOORDHOLLAND.
## OP 23 FEBRUARI 1985.
## DE KOPEREN KNOOP VAN LIMBURGSTIRUMSTRAAT AMSTERDAM.
## AANVANG 12.00 UUR.

AGENDA:

1. Opening.
2. Verantwoording van de penningmeester.
3. Verantwoording van de secretaris.
4. Verkiezing nieuwe bestuursleden.
5. Vraagstelling over bijeenkomsten op zondag.
6. De begroting voor 1985.
7. Behandeling voorstel tot wijziging van statuten.
8. Rondvraag.
9. Sluiting.

Er worden kandidaten gezocht voor de functie van voorzitter en penningmeester. Voorstellen voor de agenda en kandidaat stelling gaarne voor 2 februari. Na deze datum verschijnt een speciale ERROR 94.

ALGEMENE LEDENVERGADERING VAN DE REGIO DELFT.
Op 4 FEBRUARI 1985
E-KAFE GEBOUW ELEKTRO TH WIJK DELFT
AANVANG 20.15 UUR

AGENDA:

1. Opening.
2. Verslag kascommisie over 1984.
3. Verkiezing nieuwe bestuursleden.
4. Wat verder ter tafel komt.
5. Sluiting.

---

ALGEMENE LEDENVERGADERING VAN DE REGIO NOORD.
OP 14 FEBRUARI 1985.
MUNSTERHOES MUNSTERHEERD GRONINGEN
AANVANG 19.00 UUR

AGENDA:

1. Opening
2. Ingekomen stukken.
3. Verslag secretaris.
4. Verslag penningmeester.
5. Verslag van federatieve vergadering.
6. Begroting penningmeester.
7. Verslag kascommissie.
8. Kandidaatstelling bestuursleden
9. Korte pauze.
10. Verkiezing nieuw bestuur.
11. Rondvraag.
12. Sluiting

Na het officiele gedeelte wordt nog een videofilm vertoond over de werking van matrixen.

---

ALGEMENE LEDENVERGADERING VAN DE REGIO BRABANT OOST.
OP 12 FEBRUARI 1985.
DE WERF V/D WERFFSTRAAT EINDHOVEN.
AANVANG 20.00 UUR

AGENDA:

1. Opening door de voorzitter.
2. Ingekomen stukken.
3. Mededelingen van het bestuur.
4. Jaarverslag 1984.
5. Balans - Staat van baten en lasten.
6. Verslag kascommissie.
7. Decharge van het bestuur over het in 1984 gevoerde beleid.
8. Verkiezing kascommissie
9. Begroting 1985.
10. Statuten wijziging federatie en de regio.
11. Rondvraag
12. Sluiting.

De Cursor nr. 6 (brabant)
Piechart; P-charme en Joystick aanpassingen; Vragenspel; programma Tekenen; artikel over Arrays; Programma schema; INFO-3KB is een COS uitbreiding; Tip voor DBASE V2.0; Hardware uitbreiding voor beter beeldscherm; Ombouwen schakelkaart

De Stacker december (noord)
Decados; Wordpack relocaten; De GRAF-kaart; Scan-converting; programma tube; Infoco; Iets over de via; De ready status van de doscontroller.

Atomix nr. 14 (delft)
Wisselen van beeldscherms; Backtracking; Asselection en meerasselection; 4K ram voor de AXR-ATOM 64k kaart.

Datacheck nr. 7 (den haag)
DOSinitialisatie; Hardware fast/stop print; Karaktervergroting voor AVT printer; Insert-functie; Regressie; Programma listing; Statement dump en copy; Muvar; Grote letters op de microline; P-charme schakelsoft; Color screendump; Speech synthesizer; Shuart led display; Het CX-systeem

ROM nr. 6 (rotterdam)
Het CX-systeem; Een modem verhaal; Een voedingsverbetering

Tempus Fugit december (arnhem)
4K ram op #A000; P-charme statements CAS, Mode en Memo; Vervolg op artikel over klein printje

Acorntjesbrood december (twente)
Vervolg symbolische assembler; Sprite-editor en Paint-editor; Aanvulling op DB3.01; Aanvulling op Programflow Controller (beide aanvullingen reeds in dit nummer van A.N.)

Bronsgroen eikeltje nr. 8 (limburg)
City Bomber; Morse - Sein en oefenprogramma's; Verklarende woordenlijst; printen van operating systeem vectoren

REGIOSCHIJF NR. 1 1985.

| | | | | | |
|---|---|---|---|---|---|
| ANALYSE | 2900 | C2B2 | 004FC | 002 | A.N. nr.1 blz 25 |
| PLIST | 2900 | C2B2 | 004FC | 01A | A.N. nr.1 blz 34 |
| SPEECH | 2900 | C2B2 | 00BEE | 01F | A.N. nr.1 blz 14 |
| SPEAK | 2900 | C2B2 | 007CA | 02B | A.N. nr.1 blz 14 |
| CXS-1.7 | 2900 | AFAF | 00965 | 033 | A.N. nr.1 blz 30 |
| VRAGEN | 2900 | C2B2 | 00BA9 | 03D | A.N. nr.1 blz 36 |
| SCHEMA | 2900 | 2900 | 01700 | 049 | A.N. nr.1 blz 38 |
| OS-VECT | 8200 | C2B2 | 00495 | 060 | A.N. nr.1 blz 41 |
| DB3DOS | 9000 | 9000 | 02000 | 065 | A.N. nr.1 blz 48 |
| DB3COS | 9000 | 9000 | 02000 | 085 | A.N. nr.1 blz 48 |
| MUVAR | 2900 | C2B2 | 008BD | 0A5 | A.N. nr.1 blz 60 |
| PFC-AXX | A000 | A000 | 01000 | 0AE | A.N. nr.1 blz 65 |
| PFC-PCH | 8200 | 8200 | 01072 | 0BE | A.N. nr.1 blz 65 |
| CITYBOM | 2900 | AFAF | 017DC | 0CF | A.N. nr.1 blz 90 |
| TYPTALK | 2900 | C2B2 | 019E7 | 0E7 | A.N. nr.1 blz 76 |
| COLDUMP | 2900 | AFAF | 00759 | 101 | A.N. nr.1 blz 09 |

16 JAN 1985 CENTRUM DE LANTAERN
20.00 UUR UTRECHTSESTRAATWEG 4 NIEUWEGEIN.

16 JAN 1985 NOORD M van VLISSINGEN
ACHTER DE GROTE KERK 25 LEEUWARDEN.

26 JAN 1985 OV/GELD WIJKCENTRUM DE BOLDER.
DOBBE 62 ZWOLLE.

26 JAN 1985 BELGIE OXACO-CAPENBERGCENTER
10.30 UUR BORSBEEKSE STEENWEG 45. BOECHOUT (BIJ HOVE)

1 FEB 1985 LIMBURG ZAAL HUIVENEERS
19.30 UUR STATIONSSTRAAT 35 SITTARD

1 FEB 1985 DEN HAAG EXODUSKERK
20.00 UUR BERESTEINLAAN 263 DEN HAAG.

4 FEB 1985 DELFT E-KAFE
20.00 UUR GEBOUW ELEKTRO TH WIJK DELFT.

4 FEB 1985 ARNHEM t.o POSTGIRO
VELPERWEG 56-A (kelder). ARNHEM.

7 FEB 1985 BRABANT GEBOUW B.R.A.N.
20.30 UUR HEUVEL 7 VORSTENBOSCH.

8 FEB 1985 LIMBURG HEEMKUNDE MUSEUM
20.00 UUR ST. MARTINUSSTRAAT 12 BEEK.
N.B. Iedere donderdag avond wordt hier een bijeenkomst gehouden behalve op de donderdag voor de regioavond in SITTARD.

12 FEB 1985 BRABANT DE WERFF
20.30 UUR V/D WERFFSTRAAT EINDHOVEN.

13 FEB 1985 TWENTE MEPA-GEBOUW
20.00 UUR WIERDENSESTRAAT 40 ALMELO.

14 FEB 1985 NOORD MUNSTERHOES
19.30 UUR MUSTERHEERD. GRONINGEN.

14 FEB 1985 CENTRUM DE LANTAERN
20.00 UUR UTRECHTSESTRAATWEG 4 NIEUWEGEIN.

20 FEB 1985 LIMBURG CAFE "DE NOR"
GEERSTRAAT 484 HEERLEN

20 FEB 1985 NOORD M van VLISSINGEN
ACHTER DE GROTE KERK 25 LEEUWARDEN.

7 MRT 1985 BRABANT GEBOUW B.R.A.N.
20.30 UUR HEUVEL 7 VORSTENBOSCH.

8 MRT 1985 DEN HAAG EXODUSKERK
20.00 UUR BERESTEINLAAN 263 DEN HAAG.

Hier volgt een overzicht van de in het datasheets archief aanwezige datasheets; de kopie-kosten bedragen ƒ0,15 per kopie.

| datasheet | blz | kosten | omschrijving |
|---|---|---|---|
| 6502 | 10 | 1,50 | microprocessor |
| 6520 | 4 | 0,60 | PIA |
| 6522 | 10 | 1,50 | VIA |
| 6522 | 24 | 3,60 | VIA |
| 6530 | 10 | 1,50 | PIA + RAM + ROM |
| 6532 | 8 | 1,20 | PIA + RAM |
| 6545 | 10 | 1,50 | CRT-controller |
| 6847 | 12 | 1,80 | VDG |
| 2114 | 4 | 0,60 | 1k x 4 stat. RAM |
| 2114L | 4 | 0,60 | 1k x 4 stat. RAM |
| 21L14 | 5 | 0,75 | 1k x 4 stat. RAM |
| 2716 | 5 | 0,75 | 2k x 8 EPROM |
| upd2716 | 6 | 0,90 | 2k x 8 EPROM |
| 2732 | 4 | 0,60 | 4k x 8 EPROM |
| 2732A | 4 | 0,60 | 4k x 8 EPROM |
| 2764 | 4 | 0,60 | 8k x 8 EPROM |
| 27C64 | 5 | 0,75 | 8k x 8 CMOS EPROM |
| 27C128 | 5 | 0,75 | 16k x 8 CMOS EPROM |
| 6116 | 6 | 0,90 | 2k x 8 CMOS RAM |
| 6116LP | 7 | 1,05 | 2k x 8 CMOS RAM |
| 8416 | 5 | 0,75 | 2k x 8 CMOS RAM |
| upd446 | 6 | 0,90 | 2k x 8 CMOS RAM |
| upd8255A | 8 | 1,20 | PPI |
| 8255 | 11 | 1,65 | PPI |
| 8255A | 21 | 3,15 | PPI |

Verder zijn in het archief de datasheets van de volgende families logische bouwstenen aanwezig:

| | |
|---|---|
| TTL | 74.. en 74LS.. |
| Fast TTL | 74F.. |
| CMOS | 4... |
| High speed CMOS | 74HC.. |

Naast de kopie-kosten zijn ook nog portokosten verschuldigd:

| aantal blz | portokosten |
|---|---|
| 1 - 6 | 1,10 |
| 7 - 12 | 1,60 |
| 13 - 40 | 2,30 |
| 41 - 74 | 4,25 |
| 75 - | 5,25 |

De datasheets zijn te bestellen door het totale bedrag over te maken op girorekening: 3643518, t.n.v. G. Akkermans, Wikke 1, Huizen; onder vermelding van de gewenste datasheets.

Heb je interesse in datasheets die hier niet genoemd zijn, bel dan even op: 02152 - 60294.

# DE BASIC TABEL

In Atomware deel 1 is in het kort uitgelegt hoe de zoekroutine van de basicinterpreter werkt. In dit artikel wil ik deze materie nader uitleggen.
De interpretatie tabel in de basic-rom bestaat uit twee tabellen. De eerste tabel van #C000 t/m #C1DC bevat twee delen voor statements en functie's. De tweede tabel van #C1DE t/m #C22D is bestemd voor de rekenkundige interpretatie.
Het eerste deel van tabel 1 bevat de statements en het hoge byte van de start adresssen. Deze tabel loopt van #C000 t/m #C0EE en wordt in de basic zoek routine aangeroepen met #BFFF,X. Het tweede deel van tabel 1 bevat indexen en het lage byte van de start adresssen. Deze tabel loopt van #C0EF t/m #C1DC en wordt aangeroepen met #C0EE,X.
De twee delen van deze tabel zijn indentiek aan elkaar, hetgeen op de pagina hiernaast te zien is.
Wanneer bij de interpretatie in deel 1 een hoog byte wordt gevonden, d.w.z. een byte groter dan #80, dan wordt dit als het hoge byte van het startadres beschouwd, en wordt op dezelfde plaats in de andere tabel het lage byte van het startadres gevonden.
De zoekroutine van de interpreter kent twee zoeklussen. De eerste zoeklus kijkt naar een overeenkomst van het gezochte met de tabel. De tweede zoeklus kijkt naar het verschil met de tabel.
Indien een vergelijking tussen tabel en het te interpreteren statement juist in de eerste zoeklus, of onjuist in de tweede is, wordt in de andere tabel op dezelfde plaats een nieuw index gevonden voor BFFF. Hiermede wordt dan de zoekroutine in de tweede zoeklus herhaald of gestart.
In de routine van een statement uitvoering wordt bij het aanroepen van de zoekroutine het index medegegeven, zodat daar reeds bepaald wordt welk deel van de tabel moet worden afgezocht.
Omdat deze theoretische beschouwing niet in een keer duidelijk zal zijn, zal ik dit met een paar voorbeelden illustreren.
Als eerste dan het statement "END". De zoekroutine wordt aangeroepen met #7F als index, zodat het eerste wat getest wordt de letter U op #C07F is. Dit is dus niet juist en de eerste keer dat de vergelijking juist is gebeurd op #C08E. Dan wordt op #C17D, in het andere deel dus, een nieuw index gehaald. Dit index is #D0, zodat op #C0D0, de rest van het statement "END" wordt vergeleken. Wanneer op #C0D2 een hoog byte wordt gevonden, dan wordt deze in #53 opgeborgen. In het andere deel wordt dan nog het lage byte op #C1C1 gevonden en dit lage byte wordt in #52 gezet. Zodoende staat op #52/#53 het start adres voor een JMP (#0052).
Nu is dit het eenvoudigste voorbeeld, omdat er maar 1 statement bestaat die met de letter E begint. Daarom nemen we als tweede voorbeeld het basic statement SAVE. Hierbij wordt dan meteen duidelijk waar onze basic zijn snelheid aan te danken heeft.
Ook nu wordt bij het aanroepen van de zoekroutine als index #7F meegegeven. Op #C08B wordt de letter S gevonden, waardoor in het andere deel op #C17A index #D3 wordt gehaald. Op #C0D3 start een vergelijking met SGET, maar met de G loopt het al mis zodat in de andere tabel op #C1C2 een nieuw index, #D7 wordt gehaald. Daarna start een vergelijking met SPUT, maar nu loopt het op de P spaak, zodat het volgende index, #DB, wordt opgehaald om de zaak met SHUT tevergelijken. Dit gaat mis zodat op #C1CA index #90 wordt gehaald,

```
C000  <  =  > FE
C004  -  + C8
C007  #  (  !  ?  R  T  L  C
      A  P  E  G  B  F FO
C016  T FF  O CB
C01A  S CB  T  E  P CB
C020  T C3  H  E  N C3
C026  "  $ CE
C029 CE CC
C02B  $  , C5
C02E  $  &  ; 0D  , C3
C034 C5 C2
C036  > C7  = C7 C7 C7  = C7 C7
C03F C8  R C7 C7
C043  O  A FE
C046  $ C7
C048  H C9
C04A  E  N C9
C04D  N  D C7
C050 C9 C9 C9 C9
C054  N  D C9
C057  O  P C9
C05A  O  U  N  T C9
C05F  B  S C9
C062  T  R CF
C065  X  T CF
C068  E  T CF
C06B  G  E  T CF
C06F  I  N CF
C072  O  U  T CF
C076 C3 C3
C078  R  I  N  T C3
C07D  N  L  U  N  I  G  R  F  !  ?
      $  P  D  L  S  B  *  E F0
C090  A  V  E CF
C094  E  W C2
C097  O CC
C099  E  T C3
C09C  I  N  K C3
C0A0  I  S  T CA
C0A4  O  A  D CE
C0A8  N  T  I  L CC
C0AD  E  X  T CA
C0B1  F C5
C0B3  N  P  U  T CC
C0B8  O  S  U  B CB
C0BD  O  T  O CC
C0C1  E  T  U  R  N CB
C0C7  E  M C5
C0CA  U  N F1
C0CD  O  R CB
C0D0  N  D CD
C0D3  G  E  T CF
C0D7  P  U  T CF
C0DB  H  U  T CF
C0DF  P  U  T CF
C0E3  T  R CF
C0E6  U  4 CF
C0E9 C3 C4 CD C4
```

```
C0EF 36 3B 3C C0
C0F3 3F  6 DC
C0F6 50 51 52 53 54 57 4A 5A
     5F 62 65 68 6B 6F 2E
C105 18 AC 17 81          to
C109 1C BE 17 17 17 A2       step
C10F 22 1B 17 17 17 1B       then
C115 29 28 B6
C118 BF B6
C11A 2A B7 58
C11D 76 77 34 34 7C 3F
C123 4A 78
C125 38 6D 3A 64 74 5B 3E 7B 82
C12E C1 45 22 31
C132 40 4D 4D
C135 42 53
C137 15 D2                   ch
C139 15 15 BD                len
C13C 45 45 14                and
C13F  A 44 5F 4C
C143 15 15 86                rnd
C146 15 15 73                top
C149 48 15 15 15 7A --       count
C14E 15 15 02                abs
C151 15 15 29                ptr
C154 15 15 28                ext
C157 15 15 66                get
C15A 15 15 15 5B             bget
C15E 72 15 A6                fin
C161 15 15 15 A7             fout
C165 90 35
C167 E3 8F 8F 8F 34          print
C16C 94 A0 A8 AD B1 BD C1 CD E9 EA
     EB 78 97 99 D3 DF EC D0 4B
C17F 8F 8F 8F 0A             save
C183 8F AD AD                new
C186 8F F0                   do
C188 9C 8F 25                let
C18B 8F 8F 8F B2             link
C18F A4 9C 8F 51             list
C193 99 8F 8F ED             load
C197 8F 8F 8F D2             until
C19C 8F 8F 8F CD             next
C1A0 B3 66                   if
C1A2 8F 8F 8F 8F 81          input
C1A7 8F 8F 8F 8F D2          gosub
C1AC 8F B8 8F 05             goto
C1B0 CA C7 8F 8F 8F EC       return
C1B6 8F 8F 75                rem
C1B9 8F 8F 41                run
C1BC 8F 8F 57                for
C1BF 8F 8F 98                end
C1C2 D7 8F 8F E3             sget
C1C6 DB 8F 8F C5             sput
C1CA 90 8F 8F B6             shut
C1CE 8F 8F 8F 8F             bput
C1D2 E6 8F 47                ptr
C1D5 8F 8F 95                put
C1D8 EE  6 5C  F
```

zodat op #C090 uiteindelijk SAVE wordt gevonden, hetgeen resulteert in een startadres van #CF0A, welke te vinden is op #C093 en #C082. Indien er iets verkeerds wordt ingetikt zal in de tweede tabel #8F worden opgehaald, dit geldt alleen voor de statements, zodat dit resulteert in het start adres #F04B, zijnde de start van de zoekroutine in de monitor rom. Voor de functie's wordt index #15 opgehaald, waarmede het staradres, #F02E, voor de functie interpretatie in de monitor rom wordt gevonden.
De snelheid van interpretatie vloeit dus voort uit het feit, dat niet elk statement vergeleken wordt, om de start van het volgende statement te vinden. Maar inplaats hiervan wordt een index gehaald van een eventueel nog in aanmerking komend statement.
Voor TO, STEP en THEN wordt index #17 gevonden bij een verkeerde schrijf wijze, waardoor bij het hoge start byte #FF op #C017, in de zoekroutine error 174 ontstaat. Voor de vergelijkingen en de comma tussen bijvoorbeeld de variabelen geldt dat deze lang niet altijd veplicht zijn. Daarom eindigen deze tabellen met #FE zodat de zoekroutine met een RTS eindigt.

De rekenkundige tabel zit heel wat eenvoudiger inelkaar en bestaat uit drie delen. Deze ziet er als volgt uit:

```
TABEL rekenkundige bewerkingen
     - + █ :
C1DD 2D 2B 7C 3A FE
     * / % ! ? &
C1E2 2A 2F 25 21 3F 26 FE
     )
C1E9 29 FF
     =
C1EB 3D FF
     ! ? $
C1ED 21 3F 24 FF
     = ! ?
C1F1 3D 21 3F FF
     ' "
C1F5 27 22 FE

lage byte start adres.            Hoge byte start adres.
C1F8 B7 9A D3 EF EF               C212 C7 C7 C7 C7 C7
C1FD 13 5E 70 B3 9C 7B 7B         C217 C8 C8 C8 C8 C8 C8 C8
C204 78 78                        C21E C2 C2
C206 78 78                        C220 C2 C2
C208 EE  6 5C 5C                  C222 C3 C4 CD CD
C20C E5 75 7B 7B                  C226 C3 CD CD CD
C210 6F 7A                        C22A C3 C3
```

Het eerste deel bevat uit de te zoeken operator, terwijl de rest het start adres bevat. Er wordt gewoon gezocht en als het gevonden wordt, dan wordt het start adres in #52 en #53 gezet.
Indien een tabel eindigt met #FE betekent dit, dat hetgeen in de tabel staat niet veplicht was, zodat de zoekroutine met RTS wordt beeindigt indien het gezochte niet voorkomt in de tabel. #FF op het einde betekend dat het gezochte wel veplicht was en zodoende eindigt de routine in een error 174 indien het gezochte niet in de tabel is gevonden.

Het voordeel van de gebruikte tabel is de snelheid waarmede iets wordt opgezocht. Een nadeel is de extra [illegible] ruimte voor de indexen. De floating point diendeze methode niet toepast is dan ook beduidend langzamer.

In de laatste acorn nieuws stond op blz.59 een programma waar ik erg blij mee was. Dit programma geeft een color screendump voor de tandy CGP-115 printer plotter. Na het intypen en het plotten werd het resultaat zoals in fig.1 is weergegeven.Dit is een grafische dump in 4 kleuren van Winston Churchill. Ik vond de verhoudingen echter niet mooi en ben daarom wat aan het programma gaan sleutelen met als resultaat fig.2. De verhoudingen kloppen nu beter en de gehele papier breedte is nu benut. Dit door een aantal aanpassingen zoals het inbrengen van Floating point X en Y waarden waarmee de size van het geheel precies te bepalen is.
Hierbij tevens mijn dank aan de auteurs van dit programma.



<-- FIG.2 ^ FIG.1



```
 10 REM      ** ACORN ATOM CLEAR 4 COLOR-SCREENDUMP **
 20 REM PROGRAMMA VOOR DE TANDY CGP-115 PLOTTER/PRINTER
 30 REM V2.0 - BASIC
 40 REM Auteurs G.Zieleman en H.J.van de Riet
 50 REM Aangepast A.Slootweg
 60 P.$21$2$13; REM SCHERMUITVOER AF, PRINTER NAAR BEGIN REGEL
 70 P.$18; REM PRINTER OMSCHAKELEN NAAR GRAFISCHE MODE
 80 P."I"'; REM STEL DE OORSPRONG IN
 90 P."C0"'; REM KIES KLEUR 0 = ZWART
100 P."D460,0,460,-345,0,-345,0,0"'; REM TEKEN HET KADER
110 P."D461,0,461,-346,1,-346,1,1"'; REM TEKEN HET KADER
120 %X=0;%Y=0;%A=3.6
130 REM %A=WAARDE VOOR X OPHOGING
140 FOR C=1 TO 3; REM 3 KLEUREN AFWERKEN, ACHTERGRONDKLEUR (C=0)
150 REM             NIET GEBRUIKEN
160 P."H"';P."C"C'; REM TERUG NAAR DE OORSPRONG EN KLEURKEUZE
170 %X=0;%Y=0
180 REM =             BEELD-AFTASTROUTINE                         =
190 FOR S=#8000 TO #97FF
200 IF S%32=0;%X=0;%Y=%Y-1.8; REM NAAR DE VOLGENDE BEELDLIJN
210 P=?S/64&3;IFP=C GOS.a
220 %X=%X+%A
230 P=?S/16&3;IFP=C GOS.a
240 %X=%X+%A
250 P=?S/4&3;IFP=C GOS.a
260 %X=%X+%A
270 P=?S&3;IFP=C GOS.a
280 %X=%X+%A
290 NEXT S
300 NEXT C
310 P."M"0","-340'; REM RUIMTE MAKEN
320 P."A"'; REM PRINTER TERUG NAAR TEKST MODE
330 REM     EN INSTELLEN OORSPRONG OP LINKER KANTLIJN
340 P.$3; REM PRINTER AFSCHAKELEN
350 P.$6$12; REM SCHERMUITVOER WEER AAN, SCHERM WISSEN
360 END
370 REM=  SUBR.VERPLAATSEN PRINTKOP EN LIJN TREKKEN, LENGTE=A =
380a P."M"%X","%Y';P."J"%A","0'
390 RETURN
```

# CHATTERBOX

Zoals reeds uitgelegd in het artikel over speech-synthesizers in het nummer 1 jaargang 3 pagina 24 van ACORN NIEUWS hebben Jan Lernout en ik beiden een speech-synthesizer aangeschaft. Het betreft hier de zogenaamde "CHATTERBOX" die geleverd wordt door WILLIAM STUART SYSTEMS Ltd, QUARLY DOWN HOUSE, CHOLDERTON, Nr.SALISBURY, WILTSHIRE SP4 0DZ ENGLAND. Dit systeem is gebouwd leverbaar voor 49 pond of in kit voor 39 pond. Deze prijzen zijn zonder BTW en dateren reeds van september 1983 ! Het systeem is kompleet, d.w.z. het bevat de foneemgenerator SX-01, de nodige sturingslogica, een miniatuurversterkertje rond de 2283B, een luidsprekertje en een behuizing. Het spijtige van de zaak is dat er geen uitvoering beschikbaar was voor de ATOM. We hadden daarom gevraagd om een uitvoering voor een andere 6502 computer. Dat werd dan die van de BBC. Echter ook de aansluitingen voor de SPECTRUM en de UK101 werden meegeleverd zodat er informatie genoeg voorhanden was betreffende interfacing.
In het reeds vermelde artikel heb ik reeds gesproken over de SC-01 (zie ook PRACTICAL COMPUTING oktober 1982 p.116-123 en RADIO BULLETIN maart 1982 p.127-132) en daarbij gezegd dat er ook een SX-01 bestaat. Wel, deze SX-01 wordt in de CHATTERBOX gebruikt. De SX-01 en de SC-01 zijn beide foneemgenerators en hebben een analoge struktuur. Daarmee houdt echter alle gelijkenis op. De 64 fonemen zijn bij beide IC's verschillend. De SC-01 gebruikt 2 datalijnen voor pitch-instelling. De SX-01 beschikt niet over de mogelijkheid de pitch in te stellen. Het voordeel van de SX-01 is echter dat hij op 5V werkt in tegenstelling tot de SC-01 die minstens 7V nodig heeft. Dit spaart ons een extra voeding uit en bovendien hebben we nu geen problemen meer met signaalaanpassing tussen de 5 volts ATOM en de 7 volts CHATTERBOX. Laten we echter eerst de SX-01 bespreken.

## De SX-01

Ik zal hier volstaan met het samenvatten van de data-sheet meegeleverd met onze bestelling.
De SX-01 is een foneemgenerator die een processor, digitale filters, controle ROM en een pulsbreedte modulator voor het uitgangssignaal bevat. De 64 voorgeprogrammeerde fonemen kunnen opgeroepen worden door hun code (van 0 tot 63) naar de 8 data input lijnen te schrijven. Wat de 2 extra lijnen hier komen doen is mij niet duidelijk. Ze worden zeker niet voor de pitch gebruikt zoals bij de SC-01. In de data-sheet wordt er ook niet over gesproken.
De SX-01 heeft een 28 pens DIL behuizing en werkt goed bij 4.7 tot 5.3 volt. Het stroomverbruik bedraagt 90 mA (15 mA in stand-by). Alle input & output lijnen zijn LS-TTL compatibel, wat zoals reeds gezegd niet zo was bij de SC-01.

Funktie van de verschillende pennen:

| | | |
|---|---|---|
| 1: | GROUND | 0 V |
| 2:in | NRESET | een logische 0 reset het IC |
| 3,4,5,6: | | niet gebruikt? |
| 7:in | VDD | +5V |
| 8:out | SBY | een logische 1 indikeert dat de SX-01 inactief is en VDD mag verwijderd worden. |
| 9:out | NLRQ | een logische 0 indikeert dat de SX-01 klaar is voor het volgende foneem. |
| 10-18:in | D7-D0 | data input (foneem code). Data moet op de pennen aanwezig zijn 450ns voordat ALD gepulst wordt (12 is geen datalijn). |
| 19:in | SE | normaal 1. Een logische 0 selecteert de "autoload" mode : input data wordt geladen 1 microsec nadat om het even welke lijn logisch 1 wordt. |
| 20:in | NALD | een logische 0 pulse van minstens 200 ns laadt de foneemcode van de datalijnen. |
| 21:? | | |
| 22:? | TEST | ? |
| 23:in | VDS | stanby power, niet nodig als SBY=1. |
| 24:out | OUTPUT | pulse output ; vereist nog een laag-doorlaat filter om spraak te genereren. |
| 25:in | NSBY RESET | logisch 0 reset de standby logica. |
| 26:? | | |
| 27:in | XTAL1 | 3.12 MHz kristal aansluiting. |
| 28:in | XTAL2 | idem. |

De aansluitingen gemerkt met een vraagteken werden niet beschreven in de datasheet, zodat ik er niets over kan zeggen.

## Schema van de CHATTERBOX

Het schema van de CHATTERBOX vindt U in figuur 1. Hierin zijn NCS1 en NCS2 twee chip select signalen die beide laag moeten worden voor selectie. Deze zullen we gebruiken voor de adressering (zie verder). Dus als de SX-01 geselecteerd wordt zal hij data lezen aan zijn pinnen als NWR laag is (dan wordt NALD laag). Als echter RD laag is kunnen de lijnen NLRQ en SBY gelezen worden op bits 0 & 1 van de databus. Bits 2 tot 5 worden niet gebruikt terwijl bits 6 & 7 bedoeld zijn voor aansluiting van een spraakherkenner, de zogenaamde "BIG EARS". Hierover verder meer. Merk op dat de schakeling zichzelf reset na een power-up.

## Interfacing met de ATOM

Vooreerst de elektronische kant van de zaak. De CHATTERBOX heeft voor adressering slechts 1 byte nodig en kan rechtstreeks aan de Atom gehangen worden zonder daarvoor een VIA-poort te moeten gebruiken. Het volstaat om met enkele TTL poorten de 4 benodigde kontrolesignalen (2 chipselects, een lees en een schrijf signaal) te fabriceren. Als adres werd #BFFE gekozen, dat vlak onder het

schakelbyte van de schakelkaart zit. Dit byte #BFFE moet dan net zoals #BFFF buiten de bus gebracht worden. Door mij werd een schakeling gebruikt die tegelijk het gebruik van de schakelkaart mogelijk maakt. Immers de gebieden AXXX, EXXX en BFXX worden buiten de bus gebracht. Het schakelingetje kan op een klein printje ondergebracht worden in de Atom. Het schema ervan vindt U in figuur 2.

Met de CHATTERBOX werd een 20-aderige flatcable inklusief konnektor (van het aanperstype) meegeleverd. Ten behoeve van deze konnektor heb ik aan de achtezijde onder de hoofdkonnektor PL6 een 20 polige male-konnektor gelijmd. Een achteraanzicht van de extra konnektor is in figuur 3 getoond met aangeving van de aangesloten lijnen. Het printje voor de sturing is eraan bevestigd.

Een alternatief voor de hier voorgestelde aansluiting is het gebruik van PL8 binnen in de Atom. Deze metode werd door Jan Lernout toegepast. Ze vergt geen extra werk voor het dekoderingsprintje, maar anderzijds gebruikt men 1024 bytes waar er maar 1 nodig is. Bovendien zit PL8 binnen in de Atom wat nogal vervelend is wil men de CHATTERBOX aan- of afkoppelen. De kast van de Atom moet telkens open.

## Sturing van de CHHTTERBOX

De sturing is uiterst simpel. Het volstaat de code van het foneem naar #BFFE te schrijven. Men dient echter wel eerst te testen of bit D0 van #BFFE laag is ten teken van het feit dat de SX-01 klaar is voor een volgend foneem (dus dat NLRQ laag is). Indien A het uit te spreken foneem bevat dan zou een mogelijke routine er in BASIC er als volgt uit zien :

```
DO UNTIL ?#BFFE&1=0;?#BFFE=A
```

Let er wel op dat zelfs als de voor een bepaald foneem voorziene tijdsduur verstreken is en NLRQ dus laag wordt, het foneem toch nog verder blijft klinken ! Het zal pas ophouden wanneer een nieuw foneem aangeboden wordt aan de SX-01. Wenst men absoluut niets meer te horen dan zal men een van de vijf stopkommando's moeten geven.

## Fonemen voor de CHATTERBOX

De volgende fonemen zijn beschikbaar in de SX-01. Ze worden hier vermeld samen met hun mnemonic (voorgesteld door de leverancier), een voorbeeld (in het Engels) en de code die op de databus moet gebracht worden om dit foneem te activeren.

| ltr | mnem. | vb. | code | ltr | mnem. | vb. | code |
|---|---|---|---|---|---|---|---|
| a | AA | ma | 24 | o | AA | not | 24 |
|  | AE | hat | 26 |  | OH | low | 53 |
|  | AO | aught | 23 |  | AO | aught | 23 |
|  | EY | page | 20 |  | UX | come | 15 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | AR | arm | 59 | | OR | store | 58 |
| | UX | above | 15 | | UH | book | 30 |
| | AIR | air | 47 | | UW2 | food | 31 |
| b | BB2 | rib | 63 | | OW | out | 32 |
| | BB | builder | 28 | | OY | boy | 5 |
| c | KK1 | can | 42 | | UW1 | to | 22 |
| | KK2 | sky | 41 | p | PP | pig | 9 |
| | KK3 | comb | 8 | qu | KK3-WW | square | 8,46 |
| ch | CH | chuch | 50 | r | RR1 | real | 14 |
| d | DD2 | do | 33 | | RR2 | brain | 39 |
| | DD1 | could | 21 | s | SS | test | 55 |
| e | EH | end | 7 | | ZH | pause | 38 |
| | EE | see | 19 | sh | SH | ship | 37 |
| | EAR | ear | 60 | t | TT1 | part | 17 |
| | ER | letter | 51 | | TT2 | two | 13 |
| | ERR | err | 52 | th | TH | thin | 29 |
| | AY | eye | 6 | | DH1 | this | 18 |
| f | FF | food | 40 | u | UW1 | put | 22 |
| g | GG2 | got | 36 | | UX | up | 15 |
| | GG1 | guest | 61 | | UH | full | 30 |
| | GG3 | wig | 34 | | UW2 | rude | 31 |
| h | HH2 | hoe | 57 | v | VV | vest | 35 |
| | HH1 | he | 27 | w | WW | wool | 46 |
| i | IH | sit | 12 | | WH | which | 48 |
| | AY | size | 6 | x | KK1-SS | expert | 42,55 |
| | ERR | fir | 52 | y | YY1 | yes | 49 |
| j | JH | dodge | 10 | | YY2 | yes | 25 |
| k | KK3 | comb | 8 | | AY | sky | 6 |
| | KK2 | sky | 41 | z | ZZ | zoo | 43 |
| | KK1 | can | 42 | | ZH | azure | 38 |
| l | LL | lake | 45 | silence | | 10 ms | 0 |
| | EL | paddle | 62 | | | 30 ms | 1 |
| m | MM | milk | 16 | | | 50 ms | 2 |
| n | NN1 | thin | 11 | | | 100 ms | 3 |
| | NN2 | no | 56 | | | 200 ms | 4 |
| | NG | anchor | 44 | | | | |

## Voorbeelden

Met de voorgaande fonemen moeten dus woorden worden gevormd. Dit vergt een weinig oefening om de juiste fonemen achter elkaar te plaatsen. Men moet niet meer denken in termen van letters, maar van fonemen. Om de geinteresseerden al een basis te geven volgen hier een veertigtal Engelse woorden met hun presentatie in de mnemonics van de SX-01. Het spreekt vanzelf dat deze voorbeelden eventueel voor verbetering vatbaar zijn ; gebruik van andere fonemen dan de vermelde kan betere resultaten geven.
Omdat in deze tabel enkel Engelse woorden voorkomen zou men geneigd zijn te denken dat Nederlandse woorden onmogelijk zijn. Dit is niet zo ; men moet echter wel rekening houden met enkele beperkingen. Zo is de lijst van fonemen die de SX-01 kent niet volledig ; ze bevat enkel die fonemen die in het Engels courant gebruikt worden. Ondermeer de Nederlandse ch klank uit bv. schip ontbreekt. Men moet ze dan door 1 of meerdere andere fonemen benaderen. Vindt men dit

geen bezwaar dan zijn Nederlandse woorden ook mogelijk, eventueel met een Engels accent.

En dan nu de lijst :

| | |
|---|---|
| daughter | DD2-AO-TT2-ER |
| collide | KK3-UX-LL-AY-DD1 |
| sister | SS-SS-IH-SS-TT2-ER |
| clown | KK1-LL-OW-NN1 |
| square | SS-KK3-WW-AIR |
| uncle | UX-NG-KK3-EL |
| computer | KK1-UX-MM-PP-YY1-UW1-TT2-ER |
| two | TT2-UW2 |
| alarm | UX-LL-AR-MM |
| score | SS-KK3-OR |
| enough | EH-NN2-UX-FF |
| extent | EH-KK1-SS-TT2-EH-EH-NN1-TT2 |
| assistance | UX-SS-IH-SS-TT2-UX-NN1-SS |
| communication | KK1-UX-MM-YY1-UW1-NN1-IH-KK1-EY-SH-UX-NN1 |
| control | KK1-UX-NN1-TT2-RR2-AO-LL |
| data file | DD2-EY-TT2-UX-FF-AY-LL |
| disabled | DD2-IH-SS-EY-BB-UX-LL-DD2 |
| eight | EY-TT1 |
| fantastic | FF-AE-NN1-TT2-AE-SS-TT2-IH-KK2 |
| five | FF-AA-AY-VV |
| four | FF-AO-AO |
| good afternoon | GG1-UW2-DD2-AE-FF-TT2-UX-NN2-UW1-NN1 |
| good evening | GG1-UW2-DD2-EE-VV-NN2-IH-NG |
| good morning | GG1-UW2-DD2-MM-AO-NN2-IH-NG |
| goodnight | GG1-UW2-DD2-NN2-AY-YY1-TT1 |
| goodbye | GG1-UW2-DD2-BB-AY-YY1 |
| he | HH1-EE |
| help | HH1-AE-LL-PP |
| language | LL-AE-NG-GG1-WH-IH-DD2-SH |
| machine | MM-UX-SH-EE-NN1 |
| memory | MM-EH-MM-UX-RR1-YY2 |
| microcomputer | MM-AY-KK1-RR2-OH-KK1-UX-MM-PP-YY1-UW1-TT2-ER |
| microprocessor | MM-AY-KK1-RR2-OH-PP-RR2-OH-SS-AE-SS-ER |
| this is a fantas-<br>tic machine | DH2-IH-SS-IH-ZH-UX-FF-AE-NN1-TT2-AE-SS-<br>TT2-IH-KK1-MM-UX-SH-EE-NN1 |
| speech therapy | SS-PP-EE-TT1-SH-TT2-AE-RR2-UX-PP-EE |
| synthesiser | SS-IH-NN1-TT2-UX-SS-AY-ZH-ER |
| program | PP-RR2-OH-GG1-RR2-AE-MM |
| understand | UX-NN1-DD2-UX-SS-TT2-AE-NN1-DD2 |
| vocabulary | VV-UX-KK1-AE-BB-IH-UW1-LL-UX-RR2-YY1 |
| peripheral | PP-AE-RR2-IH-FF-EH-RR2-UX-LL |

Tot hier deze lijst van voorbeelden.

## Speech programma

Met behulp van dit programma dat bij de CHATTERBOX geleverd werd en voor de BBC bestemd was (door mij omgewerkt voor de Atom en uitgebreid) kan men tot maximum 40 zinnen van maximaal 50 fonemen

verwerken. Deze "strings" worden in geheugen gezet en beslaan 2k. Het is ook mogelijk dit stuk geheugen op tape of disk te zetten of ervan te laden.
Na een start met RUN vraagt het programma of er reeds een file in het geheugen zat (bijvoorbeeld omdat men met ESCAPE of BREAK uit het programma gesprongen is). Antwoorden met Y geeft direkt terug de oude file. In het andere geval vraagt het programma om een beginadres voor opslag van de strings. Let er wel op dat men minstens 2k doorlopend geheugen nodig heeft voor de strings. Men heeft nu enkele mogelijkheden : SAVE en LOAD saven en laden respektievelijk de stringfile naar of van tape (of disk). END beeindigt het programma. Met COMPOSE kan men strings editeren. Benevens de mogelijkheden die het programma zelf aangeeft na het kommando COMPOSE zijn er nog drie mogelijkheden. Alle drie deze kommando's worden door 1 letter voorgesteld. Daar er geen verwarring mogelijk is met foneemmnemonics kunnen zij door elkaar gebruikt worden.

De kommando's zijn :

D: delete foneem. Het betreffende foneem wordt gewist en al degene die er achter komen schuiven 1 plaats naar voor.
B: backwards. Hiermee gaan we terug naar het vorige foneem in deze string.
I: insert. Alle fonemen vanaf de huidige worden 1 plaats naar achter geschoven zodat op de plaats van het huidige foneem een vrije plaats komt waar een foneem tussengevoegd kan worden.

En dan nu de listing van het programma :

```
 10 REM *SPEECH EDITOR FOR BBC MICRO...JAN 83*
 20 REM *MODIFIED FOR ACORN ATOM
 30 REM BY CUYPERS FRANK (B)
 40 REM NOV 83*
 43 REM JOSBOX NODIG
 45 ON ERR G.a
 50 C=#BFFE;@=0
 60 DIM B10,LL4
 70 REM *TITLES*
 80 P.$12"ACORN ATOM SPEECH EDITOR"'
 90 FORI=1TO24;P."=";N.
100 P.''"(40 STRINGS OF 50 PHONEMES MAX.)"
105 IN.'"OLD FILE"$B
106 IF ?B=CH"Y" G.160
110 P.'';IN."STARTADRES STRINGS",E
120 P.''"please wait..."
130 FORI=E TO E+#800;?I=255;N.
140 FORI=E TO E+#800 S.50;?I=99;N.
150 REM *DEFINE MNEMONICS DISPLAY*
160 DIM MM63
170 RESTORE
180 FORI=0TO63;DIM A5;MMI=A;READ$A;N.
190 DATA" ",".","..","...","....","OY","AY","EH","KK3","PP"
200 DATA"JH","NN1","IH","TT2","RR1","UX","MM","TT1","DH1","EE"
210 DATA"EY","DD1","UW1","AO","AA","YY2","AE","HH1","BB","TH"
220 DATA"UH","UW2","OW","DD2","GG3","VV","GG2","SH","ZH","RR2"
```

```
230 DATA"FF","KK2","KK1","ZZ","NG","LL","WW","AIR","WH","YY1"
240 DATA"CH","ER","ERR","OH","DH2","SS","NN2","HH2","OR","AR"
250 DATA"EAR","GG1","EL","BB2"
260 REM*ASSEMBLE SPEECH ROUTINE*
270 P.$#15
280 FOR I=1 TO2;DIMP-1
290[
300:LL0 LDY@0
310:LL1 LDA(#80),Y
320 CMP@64
330 BCSLL2
340 LDA@1
350:LL3 BIT#BFFE
360 BNELL3
370 LDA(#80),Y
380 STA#BFFE
390 INY;BNELL1
400:LL2 LDA@1
410:LL4 BIT#BFFE;BNELL4
420 LDA@0:STA#BFFE;RTS
430]
440N.
450 P.$6
460aP.$12
470;P.'"COMPOSE,SAVE,LOAD,PLAY,END"'"(C/S/L/P/E)"
480 IN.$B
490 IF$B="C" G.b
500 IF$B="S" G.c
510 IF$B="L" G.d
520 IF$B="P" G.e
530 IF$B="E" P.$12;END
540 G.a
550 REM*SPEAK ROUTINE (PLAY)*
560eIN."SPEAK STRING NO.",W
570 IF W)40OR W(1 P.'"no such string";G.560
580 GOS.h
590 IF ?S=99 P.''"STRING EMPTY!!";G.i
600 !#80=S;LINKLL0
610 IN."AGAIN"$B;IF$B="Y" G.600
620 G.a
630 REM*COMPOSE*
640bIN.''"COMPOSE/EXAMINE STRING NO."W
650 IF W)40 OR W(1 P.'"NO SUCH STRING!!";G.b
660 P.$12;GOS.h
670 Y=0
680 P.'"TO EDIT,ENTER PHONEEM CODES IN"'"ORDER,EACH FOLLOWED"
690 P." BY (RETURN)"'"PRESS (RETURN) ALONE TO MOVE ON"
700 P.'"WHITOUT CHANGING VALUE.ENTER"'"'99'(+(RETURN)) TO"
710 P." FINISCH WORD"'"AND RETURN TO MENU."''
720 U=0
730 U=U+1;S=T+U-1;IF U=50 P.'"NO MORE STRING SPACE!";?S=99;G.i
740 P.U".DATA:"?S" "
750 IF ?S(64 P.$MM(?S)" "
760 DO P." ";U.COUNT=17
770 IN."NEW DATA "$B
780 IF $B="";IF ?S=99;F.I=S+1TO T+49;?I=0;N.;G.a
```

```
 790 IF $B="" G.730
 800 IF $B="99" ?S=99;F.I=S+1TO T+49;?I=0;N.;G.a
 810 IF $B="I" F.I=T+48TO S S.-1;I?1=?I;N.;?S=255;U=U-1;G.730
 820 IF $B="B" U=U-2;G.730
 830 IF $B="D" F.I=S TOT+48;?I=I?1;N.;T?49=255;U=U-1;G.730
 840 IF ?B>#2F AND ?B<#3A G.890
 850 I=-1
 860mI=I+1;IF $MM(I)=$B G.910
 870 IF I<63 G.m
 880 P."nonexistent"';U=U-1;G.730
 890 I=VAL$B
 900 IF I>64 P."error: code above 64 !"';U=U-1;G.730
 910 ?S=I;G.730
 920 REM*SAVE DATA TO DISK*
 930cP.'"ONLY CONSECUTIVE STRINGS"'"STARTING AT NUMBER 1 WILL"
 940 P." BE"'"SAVED"'
 945 IN.'"FILENAME"$B
 950 I=FOUT$B
 960 S=E
 970 IF?S=255 BPUT I,255;P.'"SAVE COMPLETE";SHUTI;G.a
 980 BPUT I,?S;S=S+1;G.970
 990 REM*LOAD DATA FROM TAPE*
1000dS=E
1005 IN.'"FILENAME"$B
1010 I=FIN$B
1020kL=BGETI
1030 IFL=255 P.'"LOAD COMPLETE";SHUTI;G.i
1040 ?S=L;S=S+1;G.k
1050 REM*STRING SELECTION*
1060hS=E+(W-1)*50;T=S
1070 P.''"STRING NO. "W": "
1080gIF?S=99 G.f
1090 IF?S=255 ?S=99;P.''"error : string incomplete !"'';G.f
1100 P.$MM(?S)," "
1110 S=S+1;G.g
1120fS=T;P.'';R.
```

## Het kommando SPEAK

Het zou natuurlijk nog interessanter zijn moest men in een BASIC-programma eenvoudig weg de CHATTERBOX laten spreken met een instruktie. Voor dit doel is de instruktie SPEAK bedoeld. Het gebruik van dit kommando is vrij eenvoudig. Als het geimplementeerd is volstaat het de volgende instruktie te geven (in een programma of in direct mode) :

```
  SPEAK UX,LL,AR,MM
```

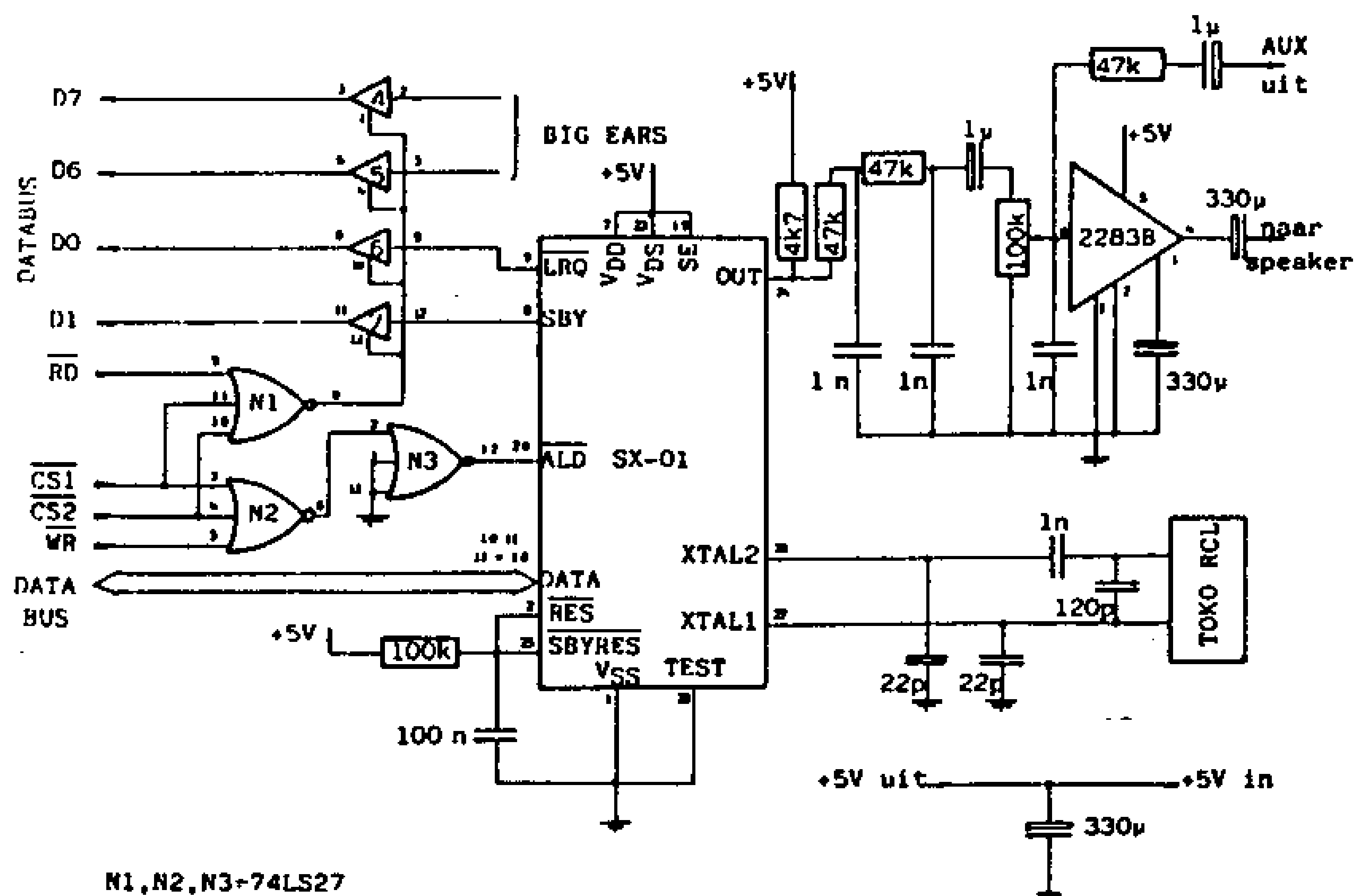
in het geval dat we "alarm" wensen te horen. Met behulp van volgend programma kan men het kommando SPEAK implementeren. Er is van uit gegaan dat de P-CHARME aanwezig is. Het programma is zodanig geschreven dat het bruikbaar is met de metode beschreven in AN 2 '84 blz.36-38 om de P-CHARME uit te breiden.

```
   10 PROGRAM SPEAK.STAT
   20 DIM SS11,LL1,TT0
   30 FOR I=0 TO 11;SSI=999;N.
   40 LL0=999;LL1=999;TT0=999
   50 PRINT$21;P=A;GOS.a
   60 P=A;GOS.a;P.$6
   70 $T="SPEAK";T=T+LEN(T)
   80 ?T=SS0/256@#80;T?1=SS0%256;T?2=#80;T=T+2
   90 A=P
  100 RESTOREb
  110 FOR I=1 TO 64
  120  READ $A
  130  ?A=?A@#80
  140  A=A+LEN(A)
  150 NEXT I
  160 ?A=#80;A=A+1;T!1=A
  170 END
  180
  190a[
  200:SS0 JSRLL1 ZOEK EERSTE NIET BLANCO
  210 STY3 BEWAAR POSITIE
  220 CMP@#2C IS HET, ?
  230 BNE SS1 NEE
  240 INY ZOEK VOLGEND KARAKTER
  250 BNESS0
  260:SS1 CMP@#3B;BEQ SS2 STOP ALS HET PUNT-KOMMA IS
  270 CMP@#D;BNESS3 STOP ALS HET #D IS
  280:SS2 JMP#C558
  290:SS3 LDA@0;STA#54 INITIALISEER FONEEMTELLER
  300 LDX@0 INIT. TABELPOINTER
  310:SS4 LDY3 INIT. KOMMANDOPOINTER
  320 LDA#54;AND@#7F;STA#54 BIT7=0 ==) 1E KARAKTER TE ONDERZOEKEN
  330 DEX;DEY
  340:SS5 INX;INY VERHOOG POINTERS
  350 LDATT0,X LAADT BYTE UIT TABEL
  360 BPLSS6
  370\1E BYTE VAN FONEEM
  380 BIT#54 WAS HET NOG 1E BYTE VAN HUIDIG FONEEM ?
  390 BMISS7 NEE,VAN VOLGEND
  400 ROL#54;SEC;ROR#54 BIT7=1 ==) 1E KARAKTER ONDERZOCHT
  410 AND@#7F;BPLSS6
  420:SS7 JSRLL1 ZOEK EERSTE NIET BLANCO
  430 CMP@#3B;BEQSS8 SPRING ALS FONEEM GEVONDEN
  440 CMP@#D;BEQSS8
  450 CMP@#2C;BEQSS8
  460:SS11 LDA#54;AND@#7F
  470 CMP@63 TABEL TEN EINDE ?
  480 BNE SS9 NEE
  490 BRK FOUT, ONBESTAAND FONEEM
  500:SS9 INC#54 VERHOOG FONEEMTELLER
  510 BNESS4 ZOEK VERDER
  520:SS6 CMP(5),Y VERGELIJK
  530 BEQSS5
  540:SS10 INX NIET GELIJK, ZOEK VOLGEND FONEEM IN TABEL
  550 LDATT0,X;BPLSS10;BMISS11
  560:SS8 LDA#BFFE
```

```
570 AND@1;BNESS8 WACHT TOT SX-01 GEDAAN HEEFT
580 LDA#54;AND@#3F STRIP BIT 7 EN 8
590 STA#BFFE SPREEK FONEEM
600 BPLSS0
610:LL0 INY
620:LL1 LDA(5),Y
630CMP@#20;BEQLL0;RTS
640:TT0;]
650 RETURN
660b
670 DATA".","..","...","....",".....","OY","AY","EH","KK3","PP"
680 DATA"JH","NN1","IH","TT2","RR1","U,","MM","TT1","DH1"
690 DATA"EE","EY","DD1","UW1","AO","AA","YY2","AE","HH1"
700 DATA"BB","TH","UH","UW2","OW","DD2","GG3","VV","GG2"
710 DATA"SH","ZH","RR2","FF","KK2","KK1","ZZ","NG","LL","WW"
720 DATA"AIR","WH","YY1","CH","ER","ERR","OH","DH2","SS","NN2"
730 DATA"HH2","OR","AR","EAR","GG1","EL","BB2"
```

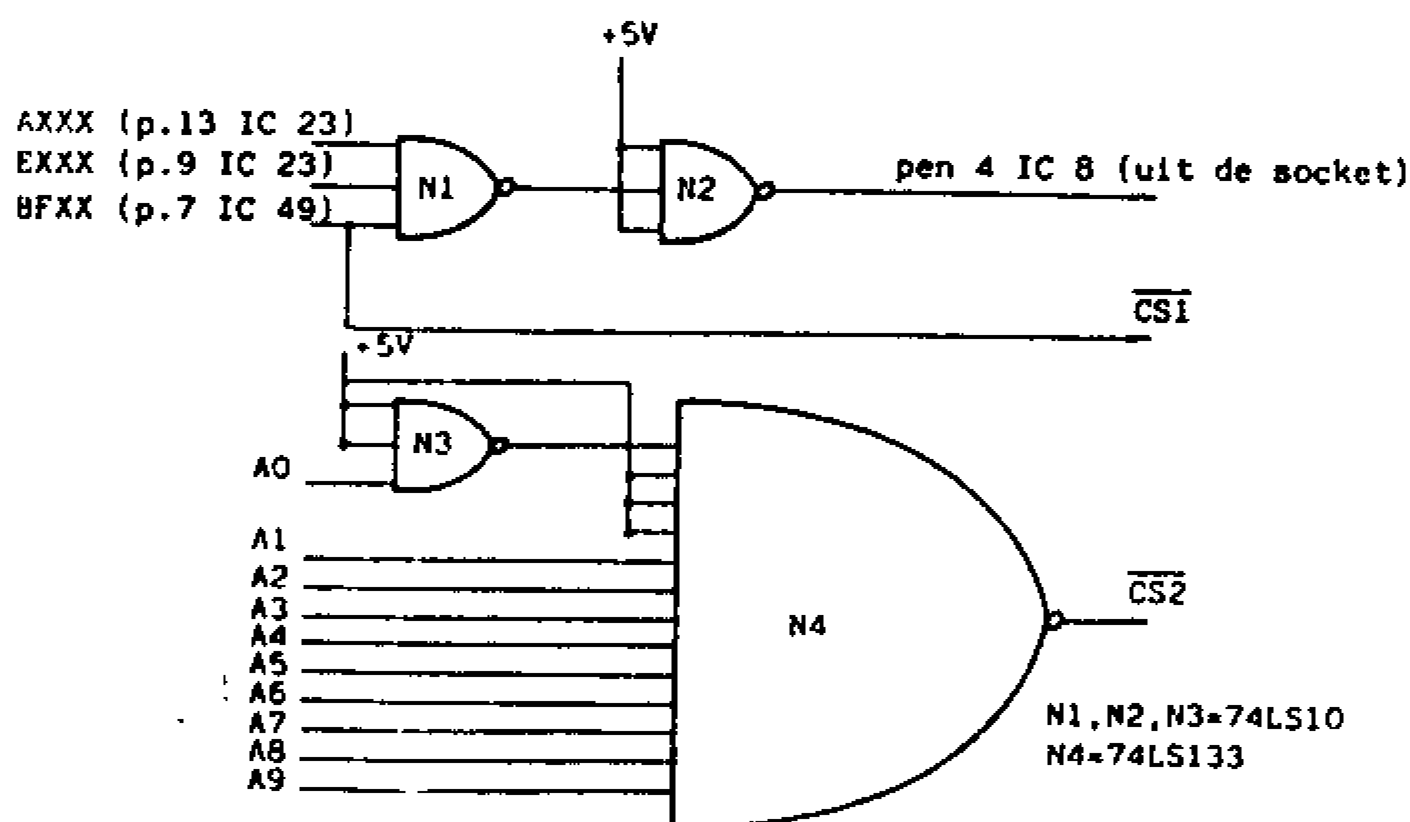
Besluit
-------

Met het hier beschreven systeem kan men zich wagen op het zeer interessante terrein van de spraakgeneratie met computer. Zoals eerder al aangehaald werd, kan men op de CHATTERBOX ook een systeem hangen voor herkenning van spraak. Dezelfde firma levert hiervoor een systeem van 49 pond, "BIG EARS" genaamd. Deze kan direkt via een plug op de CHATTERBOX aangesloten worden en vergt dus geen extra konnektie met de Atom. Helaas heb ik nog geen BIG EARS in mijn bezit en kan mij er dus nog niet over uitlaten.
Wat de spraakkwaliteit van de SX-01 betreft, wens ik nog het volgende te vermelden. De klanken geproduceerd door de SX-01 doen nogal "raar" aan. Voor iemand die het systeem kent en het al geregeld gehoord heeft, zijn de geproduceerde woorden verstaanbaar. Mijn ervaringen (o.m. op ons weekend in Rendeux) zeggen mij echter dat mensen die voor het eerst de CHATTERBOX horen nogal moeite hebben om hem te verstaan. Dit is wel een flinke beperking van de box. Doch voor beginners op het gebied van spraak mag dat niet zo'n groot nadeel zijn.
Op het moment dat dit geschreven wordt zijn er echter al weer betere foneemgenerators op de markt en voor de toekomst zullen er wellicht nog betere verschijnen. Het is dus niet onmogelijk dat binnen onafzienbare tijd met foneemgenerators dezelfde kwaliteit kan behaald worden als met woordgenerators.

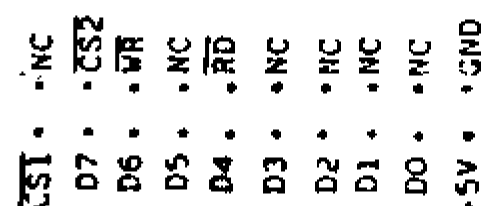



N1,N2,N3=74LS27
N4,N5,N6,N7=74LS126
TOKO RCL = L132 PC651
Luidspreker = 64Ω 0.3W

Figuur 1




Figuur 2




Figuur 3

# NOISEANALYSER

Het idee was oorspronkelijk bedoeld om met eenvoudige middelen de frekwentiekarakteristiek van recorders te kunnen testen. Dit om te kontroleren of ze voor computergebruik geschikt zijn. Het programma blijkt echter ook heel geschikt om de weergave van luidsprekers in een normale kamer te testen; iets wat anders alleen met dure apparatuur mogelijk is.
Er zijn 2 manieren om audioapparatuur door te meten: —
-1 Er wordt een zuivere toon (sinus) ingevoerd en gemeten hoeveel eruit komt. Dit wordt voor alle relevante toonhoogtes (frekwenties) gedaan en zo krijgt men de weergavekarakteristiek.
-2 Er wordt ruis in de apparatuur gevoerd. Witte ruis is een volkomen onspecifiek signaal en bevat alle frekwenties in gelijke mate. Het heet wit naar analogie met wit (zon)licht wat elektromagnetische ruis is. De karakteristiek van de apparatuur wordt bepaald door met filters kleine toonbereiken te selekteren en de sterkte van het signaal in dat bereik te meten.
Deze filters hebben elk meestal een bereik van 3 tonen ("terts") b.v. van 1 KHz tot 1,2 KHz.
In de recente elektuurs staat een zelfbouwprojekt voor dergelijke apparatuur (tertsanalyser).
Het werken met sinussen bij luidsprekers is bijzonder lastig omdat in een normale kamer door reflekties tegen muren er voor iedere frekwentie specifieke staande golven ontstaan die de meting sterk plaatsafhankelijk en dus onbetrouwbaar maken. Bij gebruik van ruis speelt dit probleem veel minder.
De NOISE ANALYSER maakt ook gebruik van ruis.
Het enige waar het programma naar kijkt zijn de nuldoorgangen. Er wordt geen gebruik gemaakt van een A/D converter. Het te onderzoeken signaal wordt op de cassette-ingang gezet en zo omgezet in Ø en 1 nivo's.
De enige informatie die er dan nog inzit is de tijd tussen twee Ø/1 en 1/Ø overgangen. Deze periodetijden zijn voor witte ruis binnen zekere grenzen toevallig. Is de ruis "gekleurd" door de karakteristiek van de apparatuur dan zal dat tot uiting komen in de gemeten waarden voor de periodetijden. Hieruit zou de karakteristiek van de app. af te leiden zijn. Helemaal zuivere koffie is dit niet; denk maar eens aan een smalle puls met een frekwentie van b.v. 1 Hz. Dit signaal bevat de frekwenties 1, 2, 3 enz. De analyser zal 2 frekwenties zien nl. een lage (tijd tussen pulsen), en een hoge (tijd v/d puls zelf). Foute boel dus, maar bedenk dat dit geen ruisachtig signaal is.
Bij ruis gaat het om het gemiddelde van een groot aantal verschillende tijden en daar blijkt de analyser wel met goede meetresultaten te komen.

HET PROGRAMMA

Fig.1 is de flowchart van de meet en sorteer routines van de NOISE ANALYSER.
Het programma meet de periodetijd van 256 achtereenvolgende overgangen en slaat die metingen op in array M. daarna worden de

metingen gesorteerd in groepen van periodetijden ofwel frekwentiegebieden waarvan er ongeveer 4 per oktaaf (1:2) zijn.
Deze z.g. logarithmische indeling past het beste bij de waarneming van ons gehoor. De limietwaarden van elke groep staan in array L. en de resultaten komen in array S.
De waarden in S worden bij eerdere metingen opgeteld in array TT waardoor de totaalmeting steeds nauwkeuriger wordt. Na iedere meetserie wordt het nieuwe totaal grafisch weergegeven. hiervoor wordt de waarde voor elk frekwentiegebied vermenigvuldigd met een correctiefactor.
Dit is nodig omdat de gebieden voor de laagste frekwenties veel kleiner zijn dan voor de hoogste en daardoor minder ruis bevatten.
Ook is het mogelijk de metingen te ijken door de correctiefactoren aan te passen.
Het meetgedeelte moet uiteraard in assembler vanwege de hoge frekwenties en dus korte meettijden. Het sorteren zou in Basic kunnen maar vergt naar schatting wel enkele seconden wat toch hinderlijk is. Het is trouwens toch niet moeilijk in assembler.

GEBRUIK

De NOISE ANALYSER is bruikbaar van 0.7 tot 12 KHz.
De hoogste frekwentie wordt bepaald door de snelheid van de meetlus in het programma. Een lagere ondergrens is mogelijk door vertraging van het meetgedeelte en berekening van andere limieten. Door tijdgebrek is dit er niet van gekomen. Bovendien moet in dat geval condensator C9 in de cassette-ingang groter worden (b.v. 220n) om de laagste frekwenties door te laten.
Voor de metingen heeft men een ruisbron nodig.
Hiervoor kan men het programma uit het manual nemen.
Er zitten wat fouten in, de verbeterde (en snellere) listing is gegeven.
Volgens het manual is het witte ruis, wat ik betwijfel.
In een van de laatste nummers van Elektuur staat ook een ruisgenerator (hardware).
Ik heb het verbeterde programma uit het manual gebruikt. Aan te bevelen is een condensator van 330 nF over de cassette-uitgang van de ATOM te zetten voor een betere verhouding hoge/lage tonen en het programma te ijken.
Hiervoor heeft men een goede recorder nodig.
Laat het ruisprogramma lopen en maak een opname van de cassette-uitgang van de ATOM. Niet te hard opnemen maar wel zorgen voor een royaal uitgangssignaal naar de ATOM.
Start de NOISE ANALYSER met RUN. Er zijn nu 3 opties:
-1 standaardwaarden: dit zijn de theoretische correcties voor witte ruis. De eerste keer altijd deze optie gebruiken.
-2 normeren: uit de voorgaande meting worden correcties berekend. Een nieuwe meting met dezelfde recorder moet nu een ongeveer horizontale grafiek geven. De Analyser is nu geijkt en er kan eventueel een andere recorder aangesloten worden.
-3 oude waarden: de correcties worden niet veranderd.
Na optie 2 altijd deze optie kiezen.
Opm: Het programma kan geijkt op band/disk gezet worden met *SAVE N-ANAL 2900 3000, opnieuw gebruiken met *LOAD N-ANAL OLD RUN EN OPTIE 3

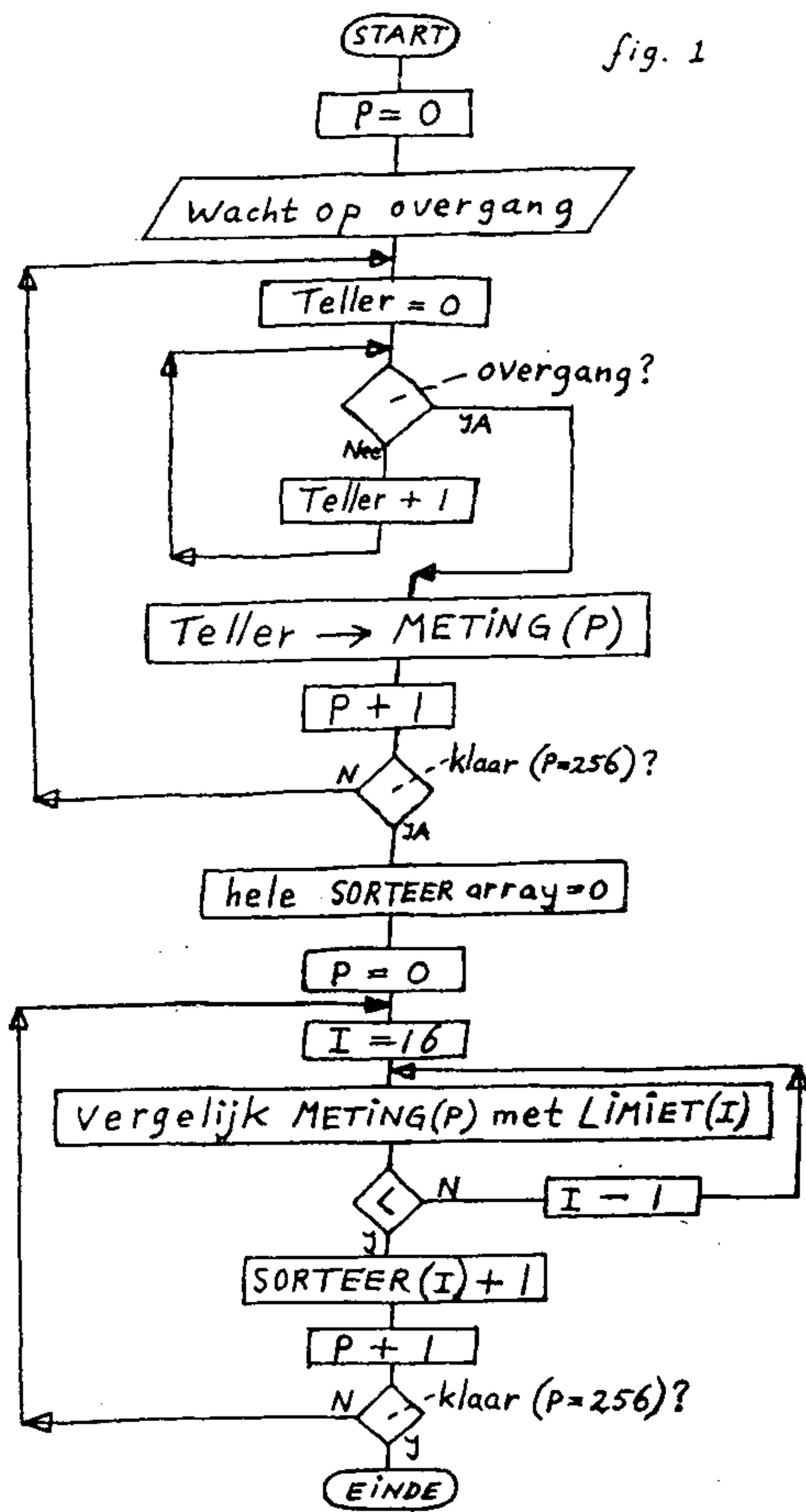
Na de keuze wacht de computer telkens op een toets om een volgende serie metingen uit te voeren en grafisch weer te geven. De vertikale lijnen geven de frekwenties 1, 2, 4 en 8 KHz aan. Als een grafiek voldoende hoogte heeft bereikt kan men een escape geven en met RUN een nieuwe meting doen en/of normeren.
Op de randen van de grafiek zijn de meetwaarden die onder resp. boven het meetbereik vielen weergegeven.

## DOORMETEN VAN LUIDSPREKERS

Probeer hiervoor op een of andere manier aan een zo goed mogelijke microfoon te komen. Maak een bandopname zoals beschreven en normeer deze. Laat het ruisprogramma lopen en sluit de cassette-uitgang aan op de versterker van de luidspreker(s). Maak nu met de microfoon een opname.
Verander de microfoonpositie voorzichtig tijdens de opname om het effekt van een specifieke positie te vermijden.
Sluit nu de recorder op de ATOM aan en meet de opname met de NOISE ANALYSER. Men heeft dan de frekw.karakteristiek van versterker en luidspreker(s). Ik heb het uitgeprobeerd op mijn 3-weg luidsprekers en de weergave aanzienlijk kunnen verbeteren door correctie van de cross-over filters. Dit was mij bij eerdere metingen m.b.v. toongenerator en scoop niet gelukt.
Het programma gebruikt de P-charme voor de READ/DATA statements.Andere boxen zijn ook bruikbaar.
Ik heb niet zoveel vertrouwen in de standaard correcties.
Het zou nuttiger zijn de DATA-regels hiervoor te vullen met de correcties na normering met een goede recorder.
Als iemand over een onberispelijke recorder meent te beschikken, wil die dan na normeren van het ruisprogramma (incl. condensator 330 nF) de correcties %CC0 t/m %CC16 uitprinten/opschrijven en aan mij toesturen.?

```
 10 PROGRAM NOISE
 20 DIM NN0,P(-1)
 30 L=#80;C=#B002
 40[
 50:NN0 TAX;AND@5;STA C
 60 TXA;AND@#60;ADC@#20
 70 ASLA;ASLA
 80 ROL L;TXA;ROLA
 90 JMP NN0
100];LINK NN0
110 END
```

```
10 PROGRAM NOISE-ANALYSER
20
30 FDIM%CC16
40 DIM M255,L16,S16,V3,TT16,LL9,P(-1)
50 FORN=0TO9;LLN=P;N.
60 P.$21;Q=P
70 GOS.a;P=Q;GOS.a
80 P.$6;G.320
90
100a[
110:LL0 \METING PERIODETIJDEN
120 LDY@0;LDA#B002;STA#80
130:LL1 LDA#B002;EOR#80;AND@#20
140 BEQ LL1;LDX@2
150:LL2 DEX;BNE LL2
160:LL3 LDX@0;LDA#80;EOR@#20;STA#80
170:LL4 INX;LDA#B002;EOR#80;AND@#20
180 BEQ LL4;TXA;STA M,Y
190 INY;BNE LL3
200 RTS
210:LL5 \SORTEREN METINGEN
220 LDY@16;LDA@0
230:LL6 STA S,Y;DEY;BPL LL6
240 LDY@0
250:LL7 LDX@16
260:LL8 LDA L,X;CMP M,Y;BPL LL9
270 DEX;BNE LL8
280:LL9 INC S,X
290 INY;BNE LL7
300 RTS
310];R.
320 REMinlezen limieten en correcties
330 DATA 1,2,3,4,5,6,8,10,12,15,19
340 DATA 23,28,34,41,49
350 DATA 1.0,2.774,2.5,2.236,2.041,1.925
360 DATA 1.581,1.491,1.525,1.302,1.06
370 DATA 1.25,1.085,0.917,0.747,0.58,0.3
380 FORN=16TO1S.-1;READ A;L?N=A;N.
390 P.$12"1-STANDAARD CORRECTIE"'
400 P."2-NORMEREN"'"3-OUDE WAARDEN"'
410 IN.'"KEUZE"$V;W=VAL$V
420 IFW=1;FORN=0TO16;READ %I;%CCN=%I*0.2;N.
430 IFW=2;FORN=0TO16;%CCN=60/TTN;N.
440 FORN=0TO16;TTN=0;N.
450 CLEAR4
460 MOVE40,0;DRAW40,191
470 MOVE100,0;DRAW100,191
480 MOVE160,0;DRAW160,191
490 MOVE224,0;DRAW224,191
500 DO LINK#FFE3
510 LINK LL0;LINK LL5
520 TT0=TT0+S?0
530 MOVE0,%(TT0*%CC0+0.5)
540 FORN=1 TO 16
550  TTN=TTN+S?N
560  DRAW(16*N),%(TTN*%CCN+0.5)
570 N.;UNTIL 0
```


START
fig. 1
P = 0
Wacht op overgang
Teller = 0
overgang?
JA
Nee
Teller + 1
Teller → METING (P)
P + 1
N
klaar (P=256)?
JA
hele SORTEER array=0
P = 0
I = 16
Vergelijk METING(P) met LIMIET(I)
N
I − 1
J
SORTEER(I) + 1
P + 1
N
klaar (P=256)?
J
EINDE

# CX-SCHAKELSOFT

Frans van Hoesel heeft de nieuwe P-Charme box in eerste instantie bedoeld voor gebruikers zonder schakelkaart. Omdat de breakvector naar deze box gezet wordt zal P-charme niet weggeschakeld mogen worden tijdens een programma.
Om nu deze box toch in een schakelsysteem te gebruiken is het onderstaande stukje schakelsoft ontwikkeld. Het is gebaseerd op de soft van Bram Poot, en is aangepast zodanig dat de P-Charme als eerste doorlopen wordt, dit omdat deze box een zeer zinvolle basic-uitbreiding vormt die niet door schakelen vertraagd mag worden. Tevens is het mogelijk om zelf ontworpen functies toe te voegen, omdat P-Fun van v.Hoesel is ingebouwd.
Deze routine is ontworpen voor het CX-schakelsysteem maar kan in principe op elke Atom draaien, mits de floating point rom m.b.v. de driewegwissel naar deze routine wijst (bvb. op de geheugenkaart).

Beschrijving van de werking.

Bij binnenkomst op #1002 wordt eerst de "lock-feature" bekeken, evt. een statement tabel op het 1000 blok doorlopen, waarna de P-charme adressen geinitieerd worden. Dit houdt in dat m.b.v. de indirecte sprong in de P-charme er via een adres in het 1000-blok, hier #1100, terug in de schakelsoft gesprongen gaat worden (zie handleiding P-charme pag. 37). Wordt er bij interpretatie door P-charme een fout ontdekt, dan wordt er teruggesprongen naar de schakelsoft.
Hier wordt gekeken of het een onbekend statement betreft (error 94), zo ja dan Poot-soft in, als het een onbekende functie betreft: P-Fun in, zo nee dan verder in de P-Charme.
In de Poot-soft wordt eerst het stuk #90-9F bewaard voor de error aanwijs-routine e.d. waarna de overige boxen afgezocht worden, als laatste voet die voor de P-charme.
Indien het statement nog steeds niet is herkend kan er nog een extra tabel doorlopen worden op een nader te bepalen plaats, indien geen gebruik van de mogelijkheid in P-charme gewenst is.
Is het statement dan nog niet herkent dan wordt de P-charme weer voorgezet, wordt het stuk #90-9F weer teruggezet en wordt een error gegeven, eventueel met aanwijzing.
Indien er op een bepaalde box "gelockt" is, bit 6 van het schakelbyte is dan gezet, wordt er over de gehele schakelroutine heen gesprongen, rechtstreeks deze box in. Dit gebeurt na het lezen van de tabel op #1000 zodat deze statements nog wel bereikt worden. Het schakelen kan weer geactiveerd worden door een (shift-return) te geven. Zie ook de beschrijving bij de routine van Bram Poot.

Mogelijkheden.

In principe kunnen nu alle statements door elkaar in een programma gebruikt worden (ook in procedures en functions), mits ze geen blijvende zero-page gebruiken in het stuk #90-9F.
Tevens kan na de eerste run van een "program" met een "goto" in het programma gesprongen worden (niet altijd!).

Bovendien moet het mogelijk zijn een miniassembler hierachter te hangen of meerletterige variabelennamen. Een functie (bv. TRUE) kan in deze routine niet in combinatie met een andere box dan P-Charme gebruikt worden (bv. DISAS(TRUE)). Dit lijkt mij geen onoverkomelijk bezwaar omdat de functie eerst aan een hulpvariabele gegeven kan worden (bv. A=TRUE;DISAS(A)).
Overigens is het zo dat de P-charme veel statements uit andere boxen overbodig maakt. EDIT en CALC herkende deze nog niet, waarvoor deze software een van de mogelijke oplossingen is. Met dank aan Henk v.d.Heijden voor het gangmaken.

```
 10REM CXS-1.7 SCHAKELSOFT
 20REM (C) DUKE URBANIK
 30REM 12-12-1984
 40
 50Q=#1000
 60!Q=0
 70?#23=0;?#24=#82
 80S=#9FFF;REM SCHAKELBYTE
 90E=#7F00;REM EVT. EXTRA TABEL
100V=7;REM VOET VAN P-CHARME
110     REM MOET LAATSTE VOET VAN
120     REM DE TE BEREIKEN BOXEN ZIJN!
130DIM BB25;P.$21
140F.I=0TO25;BBI=Q;N.
150BB7=#1100;REM INDIRECTE JUMP
160            REM ZIE REGEL 730
170            REM MOET NIEUWE PAGINA ZIJN!
180BB11=#6800;REM TABEL FUNCTIES
190F.I=1TO2;P=Q+2
200[
210\SHIFT-RETURN INGEDRUKT?
220BIT#B001;BMIBB3
230LDA@V;STA S
240:BB3
250\JMP#1103 \EVT UITBREIDEN OP #1103 E.V.
260\           TABEL EINDE: BB4
270:BB4
280LDA S;BIT S;BVSBB2
290\ INIT P-CHARME
300LDA@0;STA#3EE;LDA@#B3;STA#3FB
310LDA@BB11/256;STA#3FA
320LDA@BB7/256;STA#3EF
330LDA@V;STA#BFFF;STA S
340LDA@#AB;STA#203;LDA@#D5;STA#202
350JMP#A002 \ENTER P-CHARME
360
370
380:BB15 \VANUIT P-CHARME VIA #1100
390PLA;PLA;CMP@29;BNEBB12
400:BB13
410TAY;PLA;STA#90;JMPBB21
420:BB12 STA#00
430CMP@#5E;BEQBB14;JMP#AC4F
440:BB2 STA#BFFF
450LDA#A001;CMP@#BF;BNEBB6
```

```
460JMP#A002
470
480\ SCHAKELEN MAAR! ERROR 94
490:BB14 \SAVE #90-#9F
500LDX@#0F
510:BB10 LDA#90,X;STA#9F90,X;DEX;CPX@#FF;BNEBB10
520:BB5 \START POOT-SOS
530LDA@BB0&#FF;STA#202
540LDA@BB0/256;STA#203
550LDA@#FF;STA S
560:BB1 LDX@#FF;TXS
570INC S;LDA S
580CMP@V;BCCBB2
590:BB9
600\ NIETS GEVONDEN
610\ RELOAD #90-#9F
620LDX@#0F
630:BB8 LDA#9F90,X;STA#90,X;DEX;CPX@#FF;BNEBB8
640LDA@V;STA#BFFF
650LDA@#AC;STA#203;LDA@#4F;STA#202
660\ INDIEN GEEN TABEL OP
670\#7F00 DAN VOLGENDE 3 REGELS
680\ WISSEN EN IN PLAATS HIERVAN:
690\ JMP#C558
700LDA E;CMP@#40;BNEBB6
710JMP #10CC
720\ TABEL EINDE: #C558
730
740\ FOUT AFVANGEN
750:BB0PLA;PLA;STA#0
760CMP@#5E;BEQBB1  \SCHAKELEN
770CMP@29;BNEBB9
780PHA;LDA@7;STA#BFFF;PLA
790JMPBB13
800:BB6JMP#C558
810
820\ STARTADRES P-FUN
825\ KAN ELK ANDER ADRES ZIJN
830];P=#1A60;[
840
850\start p-fun v.hoesel
860\error 29
870:BB21;LDX @0
880:BB22;LDA #91,X;PHA;INX
890CPX @#07;BNE BB22;STY #97
900LDY @#03;STY #93;LDA @#A4
910STA #94;JSR #A8EA
920JSR #A52B;BCS BB23
930\function-call
940JMP #AB9C
950\extra-functions
960:BB23;LDA #3FA;STA #92
970JSR BB24;LDY @#03;JMP #AB82
980:BB24;LDY @#00;STY #91
990LDA #92;STA #94
1000LDA (#91),Y;STA #92;INY
```

```
1010STY #93;DEY;LDA (#93),Y
1020CMP @#E3;BNE BB25;INY
1030LDA (#93),Y;CMP @#C7
1040BNE BB25;JSR #A8EA;JMP BB24
1050\not-found
1060:BB25;LDY #97;STY #00
1070JSR #ADC6;INX;INX;TXS
1080JMP #AC60
1090]
1100REM INDIRECTE JUMP VANUIT P-CHARME
1110P=BB7;[;JMP BB15;]
1120P.$6;N.I
1130?Q=#40;?(Q+1)=#BF
1140E.
   0(C) REGIO ROTTERDAM
```

```
  10 PROGRAM TAART
  20\ (C) DUKE URBANIK
  30\ (C) REGIO R'DAM
  40\ DEMO VAN EEN PIE-CHART VOOR
  50\ GEBRUIK IN CX-SYSTEEM.
  60\ LET EENS OP HET GEBRUIK
  70\ VAN DE VERSCHILLENDE BOXEN
  80\ DOOR ELKAAR.
  90\(GAGSROM,CX-1.7,CX-2.7,CX-3.0,P-CHARME)
 100
 110TIME"00:00:00"
 120DTIME
 130COF
 140BASE#20
 150CREATE/P:2 #F,0,#F,0,#F,0,#F,0
 160CREATE/P:3#55,#A,#55,#A,#55,#A,#55,#A
 170GRMOD
 180LCASE
 190INITTURTLE4
 200CIRCLE2,128,96,60
 210ANGLE30
 220TMOVE6,60
 230MOVE128,96
 240TURN90
 250TMOVE6,60
 260PAINT130,100
 270MOVE128,96
 280TURN60
 290TMOVE6,60
 300PAINT120,100,2
 310MOVE128,96
 320TURN100
 330TMOVE6,60
 340PAINT120,90,3
 350TMOFF
 360INKEY A
 370VTAB3
 380P."TIJDSDUUR:"$T
 390CON
 400END
```

# P-LIST

Dit programma drukt op de printer een listing af van een BASIC-programma in het geheugen van de Atom. Dit gebeurt echter niet op de gebruikelijke wijze, d.w.z. via een 'handshake'-procedure, maar op interruptbasis: van tijd tot tijd kijkt de routine of de printer 'ready' is om nieuwe gegevens te ontvangen. Is dit het geval, dan worden er weer wat bytes overgestuurd. Zo nee, dan gaat de Atom gewoon verder met datgene waarmee hij net bezig was. In de praktijk gaat dit zo makkelijk dat je nauwelijks merkt dat er uberhaupt iets geprint wordt!
De benodigde interrupt wordt gegenereerd door de VIA, die sowieso al voor het printen nodig was. De routine heeft verder dan ook geen extra voorzieningen nodig. Alleen dient LINK 2 op de print gelegd te zijn.
De routine maakt gebruik van de ZP-adressen #B2-#BF en #FD, zulks om de executie van andere programma's niet te verstoren. Normaal leveren deze adressen geen problemen op. DOS-gebruikers zullen echter andere adressen moeten kiezen.
Het programma is ontworpen als nieuw statement in een 'box'. Het dient als volgt te worden aangeroepen: 'commandonaam(expr)'. De naam kan uiteraard vrij gekozen worden. De expressie dient het 'high byte' te bevatten van het blok waar het betreffende programma begint. Bv. 'DRUK#82' betekent: druk op de printer de listing af van het BASIC-programma dat begint op adres #8200. De text-pointer wordt door dit commando NIET gewijzigd!
Zoals gezegd is de routine ontworpen als nieuw commando. Afhankelijk van de gebruikte interpreter of SOS kan het daarom nodig zijn, de 'RTS'-instructie op regel 140 te vervangen door 'JMP#C55B' o.i.d. Het programma kan eventueel ook aangepast worden voor directe aanroep, bv. als volgt: Verwijder de regels 62 en 64, RUN het programma en roep het vervolgens aan met: 'A=#XX ; Y=0 ; LINK Q'.
Het printen kan niet op normale wijze worden gestopt, d.w.z. met de 'ESC'-toets. De reden hiervan moge duidelijk zijn. De routine kan echter wel op een andere manier worden gestopt, namelijk door het tegelijk indrukken van de 'CTRL'- en de 'SHIFT'-toets.
Tenslotte nog enige condities op het gebruik van de routine:

- tijdens het printen mag het betreffende programma niet gewijzigd of gewist worden.
- tijdens executie kunnen geen andere gegevens naar de printer gestuurd worden.
- geen COS of DOS commando's geven zolang de routine actief is.
- tijdens executie geen andere commando's of routines activeren die een interrupt veroorzaken, zoals 'TRACE' en 'STEP'.

```
 10 REM PLIST 0.5
 20 @=0;P.$12;DIM KK17
 30 IN."WAAR CODE OPSLAAN "Q
 40 F.I=0TO17;KKI=Q;N.
 50 S=#FD;L=#B2;Z=#BB
 60 P.$21;F.I=0TO1;P=Q;[
 62:KK0 JSR#C3C8
 64LDA#52;LDY#53;BNE KK1
 70STA L+1;STY L;STY S
 80LDA(L),Y;CMP@#D;BNE KK1;INY;LDA(L),Y;BPL KK1+3
 90:KK1 JMP#FA7D SYN-ERROR
100LDA@KK2%256;STA#204
110LDA@KK2/256;STA#205
120LDA@#40;STA#B80B;LDA@#C0;STA#B80E
130LDA@#00;STA#B804;LDA@#10;STA#B805
140CLI;RTS
150\
160:KK2 LDA#B804;TXA;PHA;TYA;PHA
170BIT#B001;BVS KK3;BPL KK5
180:KK3 LDA@2;JSR#FEFB;LDY S;BEQ KK4
190DEC S;LDX Z+1;LDY Z+2;JMP KK16
200:KK4 LDA(L),Y;CMP@#D;BNE KK13;JSR KK15
210INY;LDA(L),Y;BPL KK6
220:KK5 LDA@#40;STA#B80E;BNE KK14
230:KK6 STA L+3;INY;LDA(L),Y;STA L+2;LDX@4
240:KK7 LDA@0;STA L+4,X
250:KK8 SEC;LDA L+2;SBC#CE08,X;TAY;LDA L+3;SBC#CE10,X;BCC KK9
260STA L+3;STY L+2;INC L+4,X;BNE KK8
270:KK9 DEX;BPL KK7;LDX@4
280:KK10 LDA L+4,X;BNE KK12;LDA@32;JSR KK15;DEX;BNE KK10
290:KK11 LDA L+4,X
300:KK12 ORA@#30;JSR KK15;DEX;BPL KK11;LDY@3;LDA(L),Y
310:KK13 JSR KK15;TYA;SEC;ADC L;STA L;BCC KK14;INC L+1
320:KK14 LDA@3;JSR#FEFB;PLA;TAY;PLA;TAX;PLA;RTI
330\
340:KK15 STA Z;PLA;STA Z+3;PLA;STA Z+4
350INC Z+3;BNE KK16;INC Z+4
360:KK16 BIT#B801;BPL KK17;STX Z+1;STY Z+2;INC S;BNE KK14
370:KK17 LDA Z;JSR#FF15;JMP(Z+3)
380];N.;P.$6
390 P.'"CODE STAAT VAN #"&Q"-#"&P-1'';END
400 (c) JEN,13-11-1984
```

```
   10 REM ACORN COMPUTER VRAGENSPEL
   20 REM TINY VERSHUREN
   30 REM GEBRUIK JOSBOX
   40 ?#23=0;?#24=#37;REM ARRAY RUIMTE
   50 DIM B(1001),D(20),F(12),A(1)
   60 P.$12"  * acorn computer vragenspel *"''
   70 P." DEZE PROGRAMMA'S ZIJN SAMEN TE"'
   80 P." GEBRUIKEN MET DE VRAGENBOEKJES"'
   90 P." VAN COMPUTERELECTRO VAN JUMBO."''
  100 P."  ELK BOEKJE BEVAT 1001 VRAGEN"'
  110 P."    OVER DIVERSEN ONDERWERPEN."''
  120 P."      HET PROGRAMMA LAADT DE"'
  130 P."      VRAGEN UIT EEN BOEKJE."''
  140 P."      DE NAAM VAN HET BOEKJE"'
  150 P."       STAAT OP HET SCHERM."''
  160 P."    ***** druk een toets *****  ";LINK#FFE3
  170 P.$12"KIES:"''''"1-ANTWOORDEN INVOEREN"''
  180 P."2-ANTWOORDEN VAN RECORDER"''
  190 IN."keuze"$A;FCOS
  200 IF$A="1";GOS.i;G.n
  210 IF$A="2";GOS.r
  220nP.$12;IN."WAT IS JE NAAM "$D
  230 GRMOD;S=0;T=0;@=1
  240aP.'"             GOED          FOUT"''
  250 P."         1    2    3       A     B"''
  260 P."         4    5    6       C     D"''
  270 P."         7    8    9       RETURN"''
  280 P."              0            DELETE"''
  290 P."     "$F" VRG 1 T/M "E-1,''
  300 GOS.t
  310fFOR Z=1 TO 64;P." ";N.;P.$11$11
  320 IN."             VRAAG  NR.  "$A
  330 V=VAL$A;IF V=0 GOS.s;G.a
  340 IF V<1 OR V>=E P.$13$11;G.f
  350lIN."             ANTWOORD    "$A
  360 C=?A-#40;IFC>4 OR C<1 P.$11;G.l
  370 P.$30$10$9$9$9"          "
  380 IF B?V=C P."goed";S=S+1;PLAY C8,E8,G8,C'4;G.m
  390 P."GOED              fout";PLAY D4,C#4,C4,B2
  400 N=B?V
  410 P.'';FORZ=1TO20;P.$9;N.
  420 IF N=3 OR N=4;P.$10$10
  430 IF N=2 OR N=4;P.$9$9$9$9$9
  450mT=T+1;LINK#FFE3
  460 P.$30$10$9$9"        GOED            FOUT"''''''''''''''
  470 G.f
  480tMOVE13,183;DRAW242,183;DRAW242,43;DRAW13,43;DRAW13,183
  490 MOVE14,182;DRAW241,182;DRAW241,44;DRAW14,44;DRAW14,182
  500 MOVE15,181;DRAW240,181;DRAW240,45;DRAW15,45;DRAW15,181
  510RESTORE
  520FORN=1TO23
  530READ X,Y;MOVE X,Y
  540READ X,Y;DRAW X,Y
  550N.;R.
```

```
560DATA15,62,240,62,15,61,240,61,15,165,240,165,15,164
570DATA240,164,40,160,128,160,40,138,128,138,40,114
580DATA128,114,40,90,128,90,68,68,100,68,150,68,219,68
590DATA150,160,219,160,150,138,219,138,150,114,219,114
600DATA150,90,219,90,40,160,40,90,68,160,68,68
610DATA100,160,100,68,128,160,128,90,128,160,128,90
620DATA185,160,185,114,150,160,150,68,185,160,185,114
630DATA219,160,219,68,0,0
640sP.$12"  *** D E  U I T S L A G ***"''
650 GOS.b
660 @=1;P."JE HEBT "T" VRAGEN GEHAD"'
670 P."EN "S" GOED BEANTWOORD"''
680 U=(100*S/T+5)/10
690 P."DUS JE HEBT EEN "U''
700 IF U>6 P."goed zo, "$D''
710 IF U<6 P."dat moet beter kunnen, "$D''
720 IN."DOORGAAN "$A
730 P.$12;R.
740iP.$12"INVOER ANTWOORDEN"''
750 P."1,2,3 OF 4"''"OPHOUDEN MET "".""""''
760 IN."VANAF VRAAG NR. "$A;N=VAL$A
770 IFN<1 OR N>1001 G.760
780 @=4;P.$12;E=N
790 DO
800 P.E;IN." ANTW"$A;B?E=VAL$A
810 IF $A=".";U.1;G.840
820 IF B?E>4 OR B?E<1 P.$11;G.800
825 E=E+1
830 UNTIL E>1001
840 P.$12"ANTWOORDEN OP BAND WEGSCHRIJVEN"''
850 X=#80;!X=#21C
860 IN."GEEF NAAM (BV REKENEN)"'$F
870 F?12=#D;$#21C=$F
880 X!2=#3700;X!6=#3700;X!8=#3700+E+1
890 LINK#FFDD;GOS.b
900 R.
910rP.$12;IN."NAAM ANTWOORDEN "$F
920 X=#80;!X=#21C;$#21C=$F;X?4=0
930 LINK#FFE0;GOS.b
940 E=0;DO E=E+1;U.B?E=0 OR E>1001
950 R.
960bPLAY A'4,A'4,A'4,F1,R1,G4,G4,G4,E1;R.
```

# SCHEMA

SCHEMA is een programma waarmee direkt in CLEAR4, dus zonder eerst een file aan te maken, schema's kunnen worden getekend. Aangezien er (volgens de enquete) nog een aanzienlijk percentage van de leden niet in het bezit is van een 16K-kaart is het programma gebaseerd op een normale ATOM, maar wel met gestapeld geheugen (#2800-#4000 en #8000-#A000) en P-CHARME. Verder wordt er geen gebruik gemaakt van een joystick mogelijkheid, al is deze heel eenvoudig in te bouwen.

Het programma kent schema-symbolen, lijnen en karakterstrings, die alle kunnen worden getekend, maar ook weer kunnen worden gewist. Er wordt automatisch een file bijgehouden waarin staat opgeslagen waar welke symbolen, lijnen en karakters op het scherm staan. Wanneer er veel in de tekening zou worden gewist ontstaan er veel "gaten" in deze file, wat zou betekenen dat de file al gauw buiten de geheugen-grenzen van ons computertje zou willen gaan groeien. Daarom is er een file-opruimer ingebouwd, die alle gaten uit de file verwijdert. Deze file kan op tape worden gezet, bijv. met *SAVE "TEKENING" 9800 A000, of juist van tape worden geladen, waarna gewoon verder kan worden gewerkt aan de tekening.

Om de bediening wat te vereenvoudigen wordt er gewerkt met een soort "service-balk" bovenin het beeld, waarop wordt aangegeven wat voor invoer er wordt verwacht.

HANDLEIDING:

Na RUN vraagt ATOM om "Koude of Warme start." Bij eerste opstart K intoetsen, als er al een tekening-file aanwezig is dan W. Nu verschijnt er een knipperend kruisje in beeld, dat bewogen kan worden door het indrukken van toetsen W (boven), D (rechts), X (onder) of A (links). Wanneer het kruisje zo op de juiste plaats is gezet kan er een symbool getekend worden door het indrukken van C, een lijn getrokken worden met L, of tekst worden geschreven met T.

Wanneer er voor C is gekozen wordt er gevraagd naar de symboolcode. Dit is een getal, overeenkomend met het te plaatsen symbool. Daarna moet de stand worden opgegeven, wat een cijfer is van 0 t/m 7, waarmee het symbool gedraaid en gespiegeld kan worden. Wanneer dit gebeurd is verschijnt het symbool in de juiste stand op het scherm.

Met L wordt aangegeven dat vanaf de positie van het kruisje een rechte lijn moet worden getrokken. Het eindpunt van deze lijn kan worden vastgelegd door het kruisje met de toetsen W, D, X en A naar de juiste plaats te loodsen en dan op de spatie-balk te drukken.

Met T kan tekst worden ingetypt, afgesloten met RETURN. Om de karakters redelijke afmetingen te geven t.o.v. de symbolen en om geheugen te besparen voor de karakter-tabel is er gebruik gemaakt van karakters van 3 bij 5 pixels. Geen schoonschrift dus, maar voldoende goed leesbaar.

Het programma beschikt ook over een gum. Daartoe wordt het kruisje in de buurt van het midden van een symbool, het begin- of eindpunt van een lijn, of het eerste karakter van een string gezet en vervolgens op Y (symbool uitgummen), V (lijn verwijderen) of U (string wegpoetsen) gedrukt. Het programma zoekt nu de file af naar een symbool, lijn of string in de buurt van het kruisje, laat d.m.v. een hoog pieptoontje horen dat hij deze gevonden heeft en verwijdert deze vervolgens van het scherm en uit de file. Staat het kruisje te ver van het te verwijderen objekt, dan laat de ATOM een ontevreden toontje horen en moet het kruisje wat beter op de juiste plaats gemanouvreerd worden. Wanneer er een aantal objekten gewist zijn is het

verstandig de toets F eens in te drukken, waardoor het scherm schoongemaakt wordt, de file wordt opgeruimd en de tekening opnieuw op het scherm verschijnt. Hierna kan dan weer gewoon verder worden getekend of gewist.

HULPPROGRAMMA'S:

SCHEMA maakt gebruik van een tabel waarin een kompakte beschrijving gegeven wordt van de opbouw van de diverse symbolen. Deze tabel, die dus deel uitmaakt van SCHEMA, is aangemaakt met en kan ook gewijzigd en uitgebreid worden d.m.v. het hulpprogramma SCHEMA-SYMB. Het symbool wordt in de tabel opgebouwd gedacht rond het punt X=#40,Y=#40. Van de opeenvolgende hoekpunten die tijdens het tekenen van het symbool worden tegengekomen worden de X en Y coordinaten in de tabel opgeslagen. Als een lijn zichtbaar naar een hoekpunt loopt, dan wordt bij de X-coordinaat van dit hoekpunt in de tabel #80 opgeteld. Wanneer zo alle coordinaten van de hoekpunten van het symbool opgeslagen zijn wordt het lijstje afgesloten met een byte Ø. Zo ziet de beschrijving van een condensator (zie figuur) er bijvoorbeeld uit als:

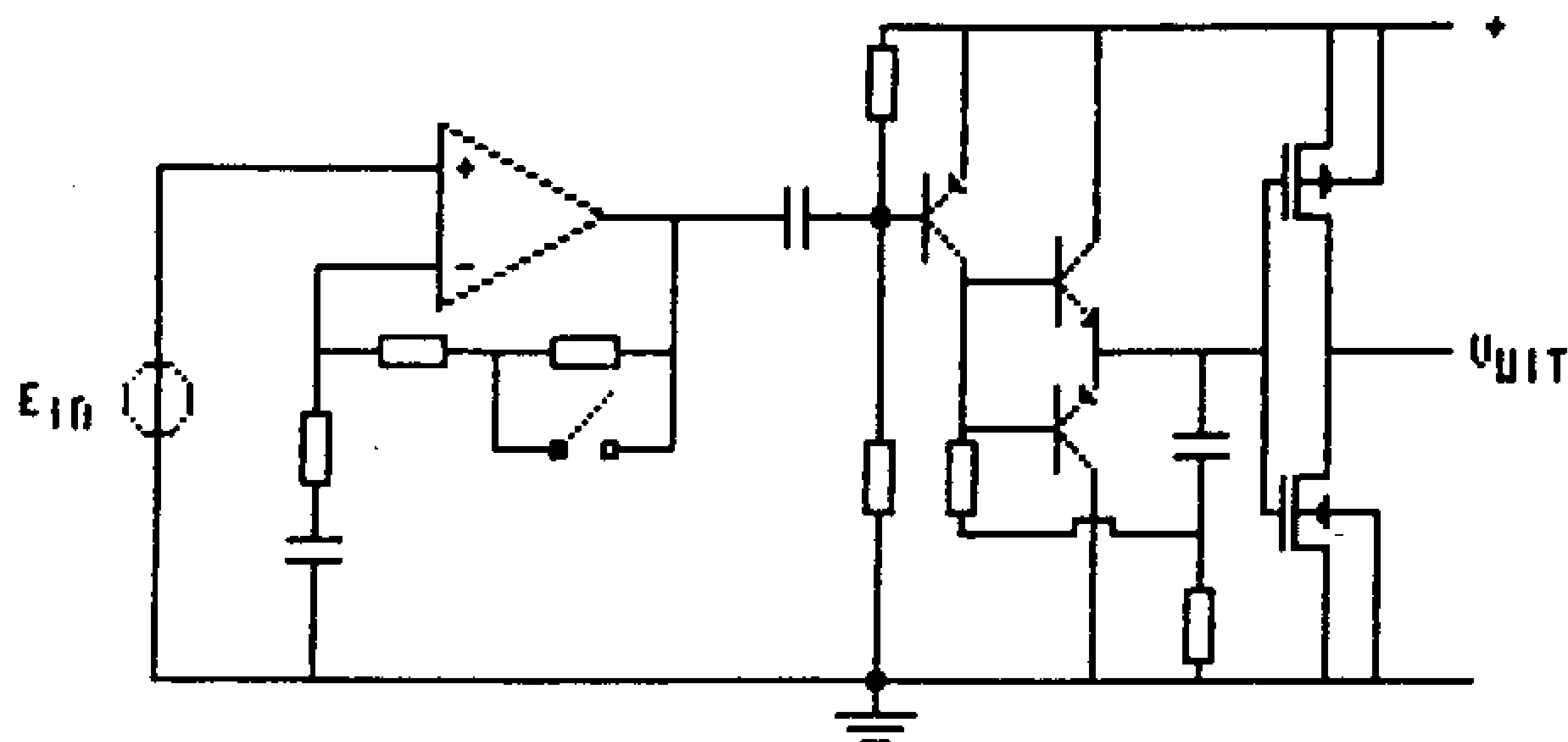
```
!T=#44BE3C3E;T!4=#44C1BC41;T!8=#40BE403B;T!12=#40C14044;T?16=0
```

waarbij de gevolgde route loopt volgens het pad: 1-2 3-4 5-6 7-8.

Volgens dit concept zijn zelf wijzigingen in of toevoegingen aan de symbooltabel aan te brengen.
Voor het schrijven van de karakters wordt door SCHEMA gebruik gemaakt van een machinetaal programma en een karakter-tabel. De bijbehorende source-kode wordt gegeven in het programma SCHEMA-CHAR.

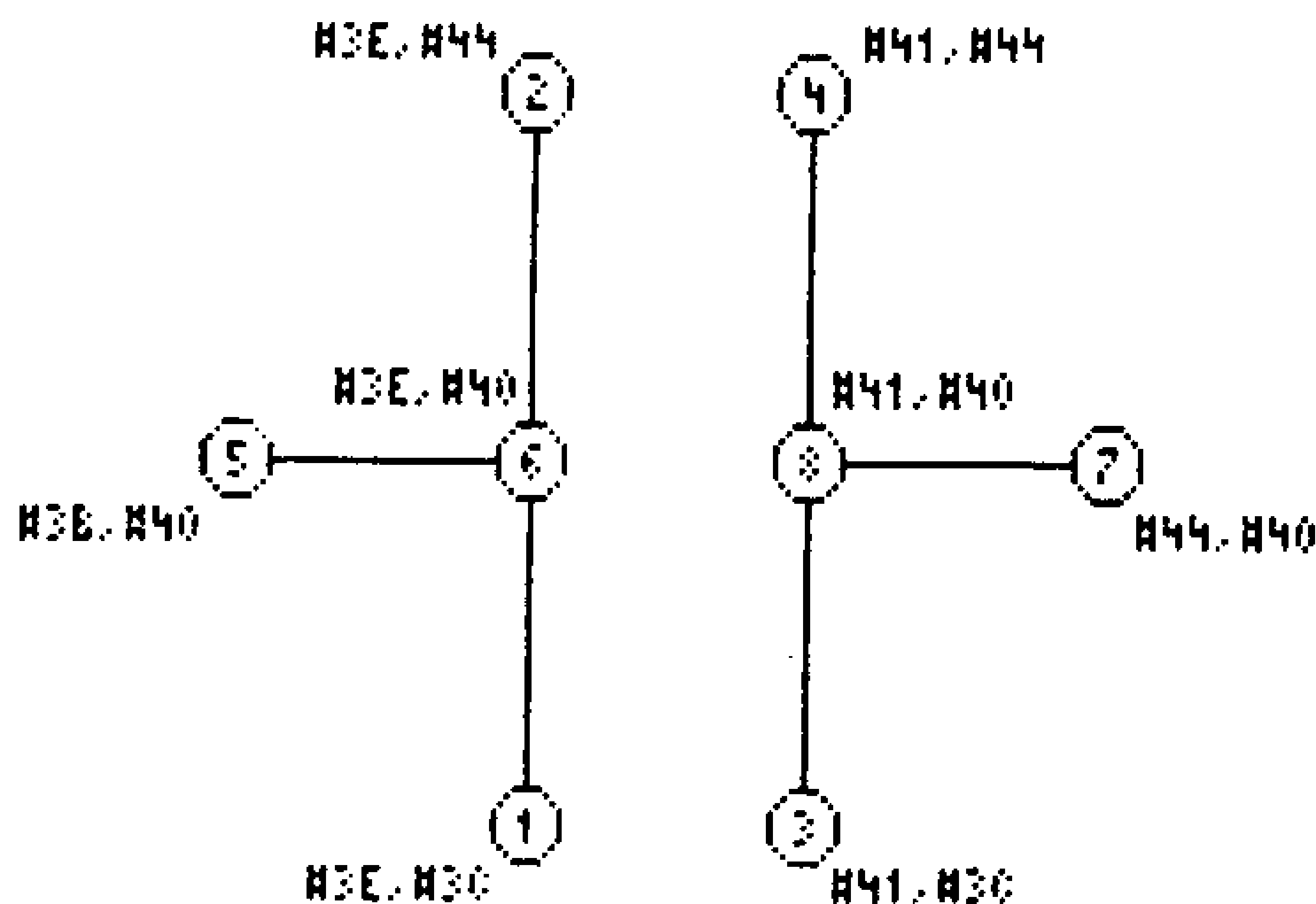
N.B.: de hulpprogramma's SCHEMA-SYMB en SCHEMA-CHAR zijn dus alleen nodig als je iets aan het programma SCHEMA zou willen wijzigen of toevoegen.

Rest nog het geven van een voorbeeldje dat met SCHEMA getekend is.




VOORBEELD VAN EEN -OVERIGENS ONZINNIGE- SCHAKELING, GETEKEND MET BEHULP VAN HET PROGRAMMA "SCHEMA".

HH



Het onderstaan programma is door mij geschreven omdat meerdere programma's gebruik maken van andere operating system vectoren.
Met dit programma is het mogelijk om alle vectoren uit te printen inclusief de orginele benaming. Het programma kan opgenomen worden in een utility rom of als source in de schakelsoft. (Aanroepen bijvoorbeeld met OSV).
De assembler neemt 154 bytes in beslag dit is inclusief 42 bytes data voor de namen van de vectoren.

```
 10REM PRINT OSV
 20REM
 30REM ************************
 40REM * DOOR CHARL DE MOOR *
 50REM *   OP GEN HOES 62    *
 60REM *      BRUNSSUM       *
 70REM * TEL:  045 - 272358 *
 80REM *    20 JULI 1984     *
 90REM ************************
100REM
110P.$6$12" OSV VECTOREN"''"ASSEMBLEREN NAAR ";IN.Q
120DIMNN(9)
130F.N=0TO9;NNN=#FFFF;N.
140P.''" pass 1"'$21;F.N=1TO2;IFN=2P.$6$11" pass 2"'''
150P=Q;[
160JSR#F7D1;]
170!P=#0D0A0A0C;P=P+4
180$P="OPERATING SYSTEM VECTORS:";P=P+LENP
190!P=#0D0A0A;P=P+3
200;[NOP
210LDX@0;STX#98;:NN1;LDY@0
220:NN0;LDANN9,Y;JSR#FFF4;CPX@02;BEQNN2;INX
230:NN3;INY;JMPNN0
240:NN2;STY#99;JSR#F7D1;]
250$P="VEC  :";P=P+LENP
260[;NOP
270INC#98;LDX#98;LDA#200,X;JSR#F802
280DEC#98;LDX#98;LDA#200,X;JSR#F802;INC#98;INC#98
290JSR#F7D1;]
300$P="    ";P=P+LENP
310[;NOP
320LDX@0;LDY#99
330CPY@#29;BEQNN4;JMPNN3
340:NN9;];REM TABEL
350$P="NMIBRKIRQCOMWRCRDCLODSAVRDRSTRBGTBPTFNDSHT";P=P+L.P
490[;:NN4;JMP#C55B
500];N.;@=0;P.'''"START : #"&Q'"EINDE : #"&P'
510P."LENGTE:  ",P-Q" BYTES"''
520P." DRUK OP EEN TOETS VOOR EEN"'" TEST VAN HET UITPRINTEN"
530P." VAN DE"'
540P." OPERATING SYSTEM VECTOREN."'" DEZE ROUTINE KAN OP"
550P."GENOMEN"'" WORDEN IN EEN TOOLKIT OF IN"'
560P." DE SCHAKEL SOFT"''
570LI.#FE94;LI.Q;E.
```

# DOSHULP

Wellicht is het volgende de meeste disk eigenaars reeds overkomen . U wist een file op diskette met *DELETE en komt daarna tot de konstatatie dat die file niet mocht gewist worden . Geen paniek echter , zolang U ondertussen nog geen andere file gesaved heeft , kan U met behulp van het hier voorgestelde programma die file volledig herstellen . Zelfs indien U reeds een nieuwe file gesaved heeft , gaat dit nog , voor zover de gewiste file niet de laatste was die op diskette stond en de gesavede file langer was dan de gewiste.
De reden waarom dit kan is simpel : als U een *DELETE doet dan wordt alleen de naam van de file uit de cataloog gewist , de data zelf blijft onaangetast . Echter bij een volgende *SAVE zal die data hoogst waarschijnlijk overschreven worden omdat de DOS niet meer "weet" dat er daar iets stond . Dat is ongeveer hetzelfde als bij NEW in basic . Het oude programma wordt ook niet gewist en kan met OLD teruggekregen worden voor zover er nog geen nieuw programma ingevoerd werd .
Wel , het onderstaande programma stelt de gebruiker in staat iedere sector op diskette apart in het geheugen te laden , onafhankelijk van de inhoud van de cataloog .
U heeft de volgende kommando's ter beschikking :

L:
--
Laadt een sector naar een buffer op £2300 . Op de vraag SECTOR? geeft U het sectornummer tussen 0 en 399 . Na uitvoering wordt dit nummer automatisch met 1 verhoogd zodat U bij een doorlopende zoekbeurt op de vraag SECTOR? gewoon return kunt geven . Initieel wordt dit nummer op 0 gezet .

X:
--
Laat de inhoud van de buffer op het scherm zien als ASCII karakters .

H:
--
Laat de inhoud van de buffer op het scherm zien als hexadecimale getallen in het formaat van een hexdump .

S:
--
Schrijft de inhoud van de buffer weg naar geheugen . De plaats van dat geheugen bepaalt U zelf door dat te specifieren bij de vraag STARTADRES? . Geeft U gewoon return dan wordt vanaf £4000 geschreven. Na iedere save wordt dit adres automatisch met £100 verhoogd .

I:
--
Geeft informatie over de stand van de sector- en adresteller . Let op dat het sectornummer gegeven wordt in twee delen : tracknummer (tussen 0 en 39) en sectornummer op die track (tussen 0 en 9).

E:
--
Stopt het programma .

A:
--

Dit kommando doet 3 dingen na elkaar . Eerst wordt de buffer naar geheugen geschreven , dan wordt de volgende sector naar de buffer geladen en vervolgens wordt ze als ASCII op het scherm gebracht . Let erop dat er niet om sectornummer of startadres gevraagd wordt . Deze moeten beide reeds met een L of S geinitialiseerd zijn .
Deze instruktie is bijzonder handig voor het achtereenvolgens laden van gewiste sectors .

B:
--

Idem als A maar nu wordt een Hdump in plaats van een Xdump gegeven .

N:
--

Idem als A maar er wordt geen data naar geheugen geschreven . Dus speciaal bedoelt voor het zoeken naar het begin van de gewiste file .

Indien U er dus voor zorgt dat de eerste regel een REM statement bevat met de naam van het programma , dan kunt U met behulp van de toets N vrij snel de sector terugvinden waar het programma begint . Met A laat U dit dan naar geheugen . Als het volledige programma in geheugen zit , stopt U met E . Daarna kunt U met een *SAVE het programma terug op de diskette plaatsen .
Dezelfde metode kan aangewend worden als om een of andere reden de kataloog op de dskette verminkt is .
En dan nu de listing van het programma . Het gebied £2300-£23FF wordt als buffer gebruikt . Het programma kan dus bijvoorbeeld naar £2400 geassembleerd worden . Het "normale" geheugengebied is dan voor het opslaan van sectors beschikbaar .

Listing:
========

```
   10REM DOSHULP
   20REM INDIVIDUEEL LEZEN VAN
   30REM SECTOREN OP DISC
   40REM
   50REM CUYPERS FRANK ACB
   60REM
   70DIMLL40,MM20,SS20,TT20
   80FORI=0TO20;LLI=-1;LL(I+20)=-1;MMI=-1;SSI=-1;TTI=-1;N.
   90IN."WAARHEEN MACHINETAAL"R
  100P.$21
  110FORI=1TO2;P=R
  120[
  130\INITIALISATIE
  140:SS1 JSR£FD69
  150LDAà0;STA£84;STA£E1;STA£85;STA£82;LDAà£40;STA£83
  160JSR£F7D1;];$P="DOS-AID DOOR CUYPERS FRANK ACB";P=P+LENP;[
  170NOP;JSR£FFED;JSR£FFED;JSR£F7D1;];$P="  DRUK EEN TOETS"
  180P=P+LENP;[;NOP;
  190:LL30 LDXà10;JSR£FB83;JSR£FE6B;LDXà13
  200:LL31 LDA£800C,X;EORà£80;STA£800C,X;DEX;BNELL31
  210JSR£F£71;BCSLL30;LDAà£80;STA£E1;LDY£E0;JSR£FD44;JSR£E000
```

```
220:SSOJSR£FFED;JSR£F7D1
230];$P="(X/H/S/L/E/A/B/I/N?)";P=P+LENP
240[;NOP;JSR£FFE3;PHA;JSR£FD69
250PLA
260CMPà£58;BNELL12;JMPLLO
270:LL12 CMPà£48;BNELL13;JMPLL3
280:LL13 CMPà£53;BNELL14;JMPLL5
290:LL14 CMPà£4C;BNELL15;JMPLL8
300:LL15 CMPà£41;BNELL17;JMPLL18
310:LL17 CMPà£42;BNELL25;JMPLL24
320:LL25 CMPà£49;BNELL27;JMPLL26
330:LL27 CMPà£4E;BNELL33;JMPLL34
340:LL33 CMPà£45;BNESSO;RTS
350\XDUMP VAN PAGINA £23
360:LLO
370LDXà0
380:LL2 LDA£2300,X;CLC;ADCà£20;CMPà£80;BCSLL1;EORà£60
390:LL1 STA£8000,X;INX;BNELL2
400LDYà0;STY£E0;STY£DE;LDAà£81;STA£DF;JSR£FD44;JMPSSO
410\ HDUMP
420:LL3 JSRLL23;JMPSSO
430:LL23 LDAà14;JSR£FFF4;LDY£E0;JSR£FD44;LDYà0;STY£E0;LDAà£81
440STA£DF;LDAà£E0;STA£DE;JSR£FD44;LDYà0;TYA
450:LL21 LDXà8;JSR£F802;LDAà£3A;JSR£FFF4
460:LL22 LDA£2300,Y;JSR£F7FA;INY;DEX;BNELL22
470JSR£FFED;TYA;BNELL21;LDAà15;JSR£FFF4;RTS
480\ SAVE ROUTINE
490:LL5 JSRLL19;JSRMMO;JMPSSO
500:LL19JSR£F7D1
510];$P="STARTADRES";P=P+LENP;[
520NOP;JSRTTO;BEQLL6;LDA£52;STA£82;LDA£53;STA£83
530:LL6 LDYà0
540:LL7 LDA£2300,Y;STA(£82),Y;INY;BNELL7;INC£83;RTS
550\PRINT GEDAAN
560:MMO JSR£F7D1
570];$P="GEDAAN";P=P+LENP;[
580NOP;RTS
590\LEES GEHEEL GETAL
600:TTO JSR£CD09;CPYà£40;BEQTT1
610LDAà£40;STA5;LDAà1;STA6;LDYà0;STY3;JSR£C3C8;LDAà1
620:TT1 RTS
630\LOAD ROUTINE
640:LL8 JSRLL16;JSR£E226;JSRMMO;JMPSSO
650:LL16 JSR£F7D1
660];$P="SECTOR";P=P+LENP;[
670NOP;JSRTTO;BEQLL9;LDXà£FF
680:LL10INX;SEC;LDA£52;TAY;SBCà10;STA£52;LDA£53;SBCà0;STA£53
690BCSLL10
700CPXà40;BCCLL32;JSR£F7D1;];$P="ALLEEN O TOT EN MET 399!!"
710P=P+LENP;[;NOP;JSR£FFED;JMPLL16
720:LL32 STX£84;STY£85
730:LL9 JSR£E75B WRITE SPECIAL REGISTER
740LDA£84;STA£EC;LDA£85;STA£ED;LDAà1;STA£F1
750LDAà10;STA£F0
760:LL39 JSR£E792;LDAà£22;STA£F7 LAADT LOAD ROUTINE
770LDAà£53;JSR£E7ED;JSR£E7A4;BNELL39
780LDAà9;INC£85;CMP£85;BCSLL11;LDYà0;STY£85;LDAà39;INC£84
```

```
790CMP£84;BCSLL11;STY£84
800:LL11 RTS
810\SAVE,LOAD&XDUMP
820:LL18 JSRLL6;JSRLL9;JSR£E226;JMPLLO
830\LOAD&XDUMP
840:LL34 JSRLL9;JSR£E226;JMPLLO
850\SAVE,LOAD&HDUMP
860:LL24 JSRLL6;JSRLL9;JSR£E226;JMPLL3
870:LL26 JSR£F7D1;];$P="SECTOR=";P=P+LENP;[
880LDA£85;STA£16;LDAà0;STA£25;STA£34;STA£43;STA£321;JSR£C589
890JSR£FFED;JSR£F7D1;];$P="TRACK=";P=P+LENP;[
900LDA£84;STA£16;JSR£C589;JSR£FFED;JSR£F7D1;]
910$P="VOLGENDE GEHEUGENPLAATS=£";P=P+LENP;[
920LDXà£82;JSR£F7F1;JSR£FFED;JMPSSO
930]
940N.;P.$6;END
```

N.B.: Er weze opgemerkt dat in AN 3 '84 reeds een gelijkaardig programma van Henk Reinders gestaan heeft .

N.B.: Deze bijdrage is zoals de anderen van mijn hand in dit nummer met behulp van de wordpack getypt op een Olivetti Praxis 35 . Dit geeft enkele problemen met karakters die niet voorzien zijn op de typemachiene . Vandaar dat £ gebruikt wordt in plaats van # en à i.p.v. @ .

```
 10 PROGRAM LOGISCHE GRAP
 20 CLEAR 4
 30 DO
 40 MOVE 0,0
 50 FOR X=256 TO 0 STEP -4
 60 FOR Y=0 TO 192
 70 DRAW X,Y
 80 N.;N.
 90 FOR A=#8000 TO #9800
100 ?A=?A:#80
110 N.
120 FOR A=#8000 TO #9800
130 ?A=?A⊠#80
140 N.
150 FOR A=#8000 TO #9800
160 ?A=?A&#80
170 N.
180 UNTIL 0
190 END
```

# DRIVE STATUS DISPLAY

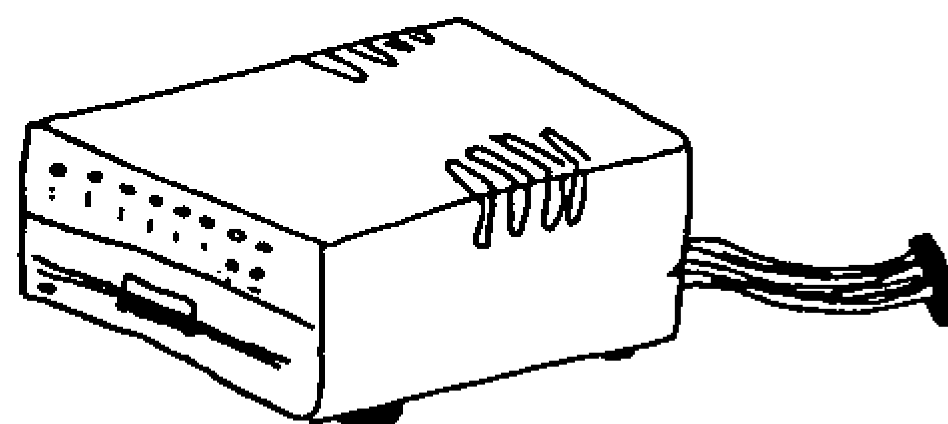
Na het aanschaffen van een slim line drive hield ik in mijn bestaande drive kast de helft aan ruimte over.
Dit kan je natuurlijk dicht maken, of je kan een kleinere kast maken. Ik hield het voor het eerste,dicht maken met een stukje aluminium plaat. In deze plaat echter zijn een aantal led's geplaatst welke de status signalen van de drive doorgeven.
In het totaal zijn er 10 led's aangebracht waarvan er 8 zijn bestemd voor de status signalen en 2 voor de +5 en +12 volt.

De 8 status signalen zijn de volgende:

INDEX, DIR, WRITE DATA, MOTOR ON, STEP, TRK Ø, WRITE PROTECT EN READ DATA.
Deze signalen zijn via een buffer (81LS96) met de drive verbonden. Dit doen we door aan de kabel die al naar uw drive gaat nog een 34 polige connector te persen. Dezelfde soort die ook al aan de kant van de controller kaart zit. Voor de +12 volt moet een apart draadje gelegd worden evenals voor de voeding van het geheel.
Voor dit shugart display is een print kaartje getekent waar al deze zaken met connector en stekers zijn te verbinden. Doordat de shugart aansluiting universeel is kan dit display ook voor andere systemen worden toegepast.

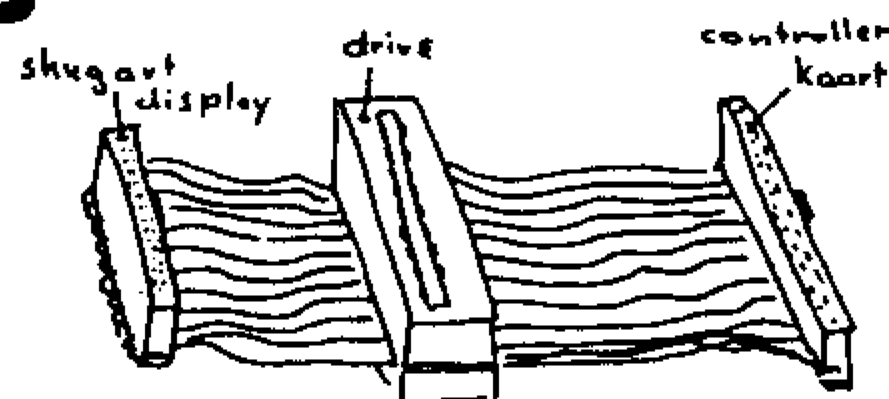
De betekenissen van de LED's zijn, van links naar rechts:

1e index
2e dir
3e write data
4e motor on
5e step
6e trkØ
7e wr prot
8e read data

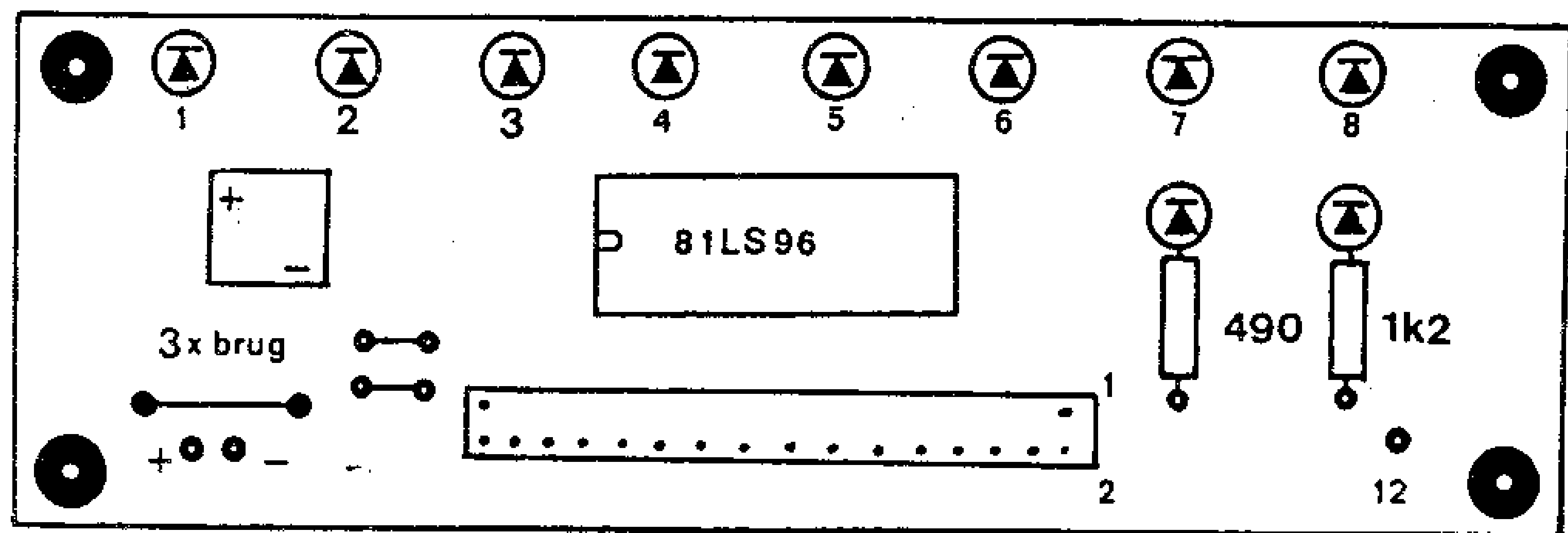


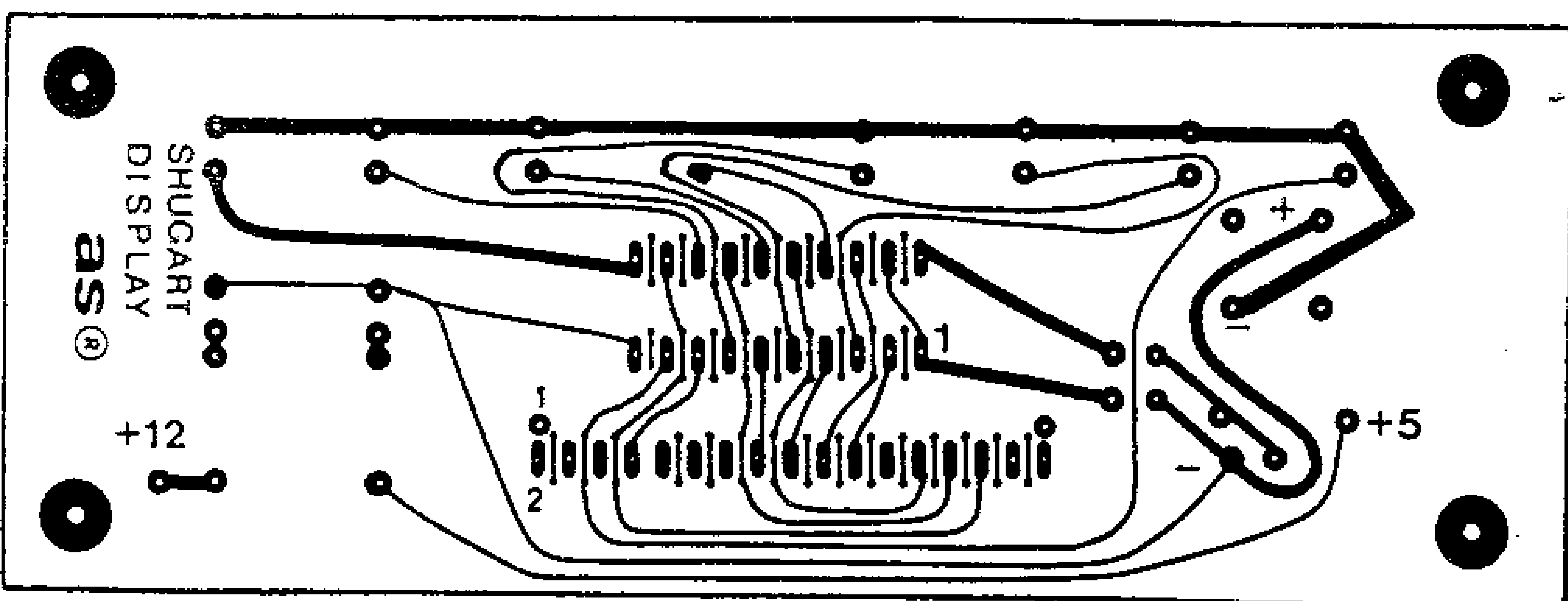
Onderdelen lijst:

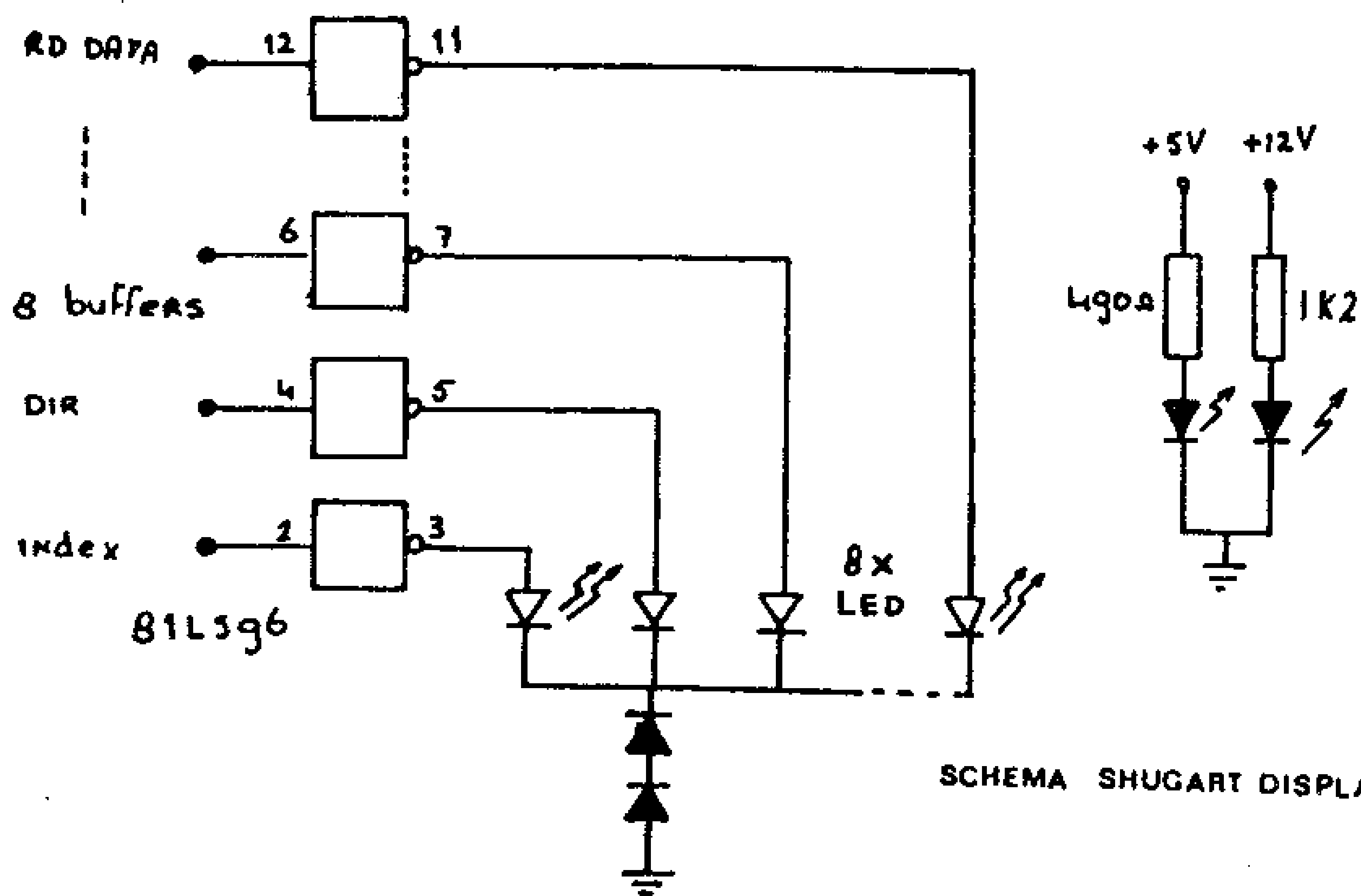
10 x LED 5mm
1 x 81LS96
1 x gelijkrichter (klein, rond/vierkant)
1 x 2Ø polige ic voet
1 x 34 polige pers connector
1 x 64 polige connector kam
1 x weerstand 12ØØ ohm
1 x weerstand 49Ø ohm




Vergeet bij het monteren van de onderdelen de drie draadbruggen niet. Monteer de led's hoog op de print i.v.m de connectors, het is ook mogelijk de leds aan de soldeer zijde te monteren, aan u de keuze. Knip van de 64 polige kam 2 enkele pinnen en 1 x 34 pinnen af. En monteer deze onder het ic.




SCHEMA SHUGART DISPLAY.

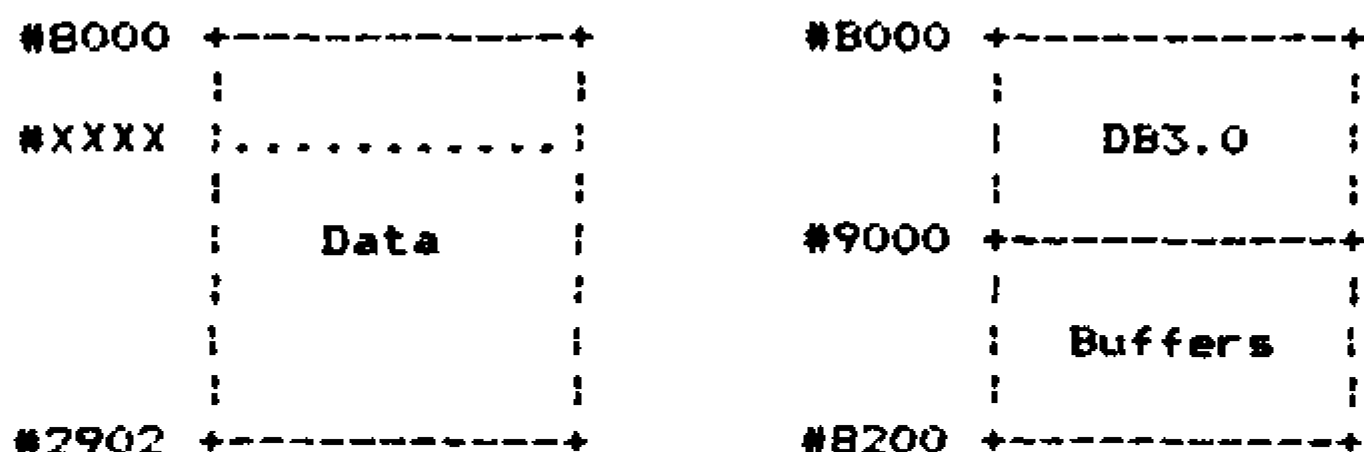
# dbase 3.00

## Inleiding.

Langzamerhand zal iedereen de toepassingsmogelijkheden van een database- of bestandsorganisatieprogramma wel kennen. Voor de ATOM zijn er al een heel stel in omloop zoals INFOMASTER, de database van ACORNSOFT, DBASE V2.0 etc. Met DB3.00 heb ik geprobeerd een programma te presenteren met redelijk wat mogelijkheden aan de ene kant en een gebruikers vriendelijke bediening aan de andere kant. Verder heb ik ernaar gestreefd om "Up-Wards" compatible te blijven met DBASE 2.0 en eerdere versies. Wat wil dit zeggen? Alle bestanden die met behulp van DBASE 2.0 zijn vervaardigd kunnen zonder meer door DB3.00 verwerkt worden, andersom echter, is dit niet mogelijk. Bestanden die een keer door DB3.00 zijn verwerkt en daarmee ook "gesaved" zijn, kunnen NIET meer door de oudere DBASE versies verwerkt worden.
Diegenen die al enige ervaring met DBASE 2.0 hebben zullen met deze nieuwe versie weinig moeilijkheden hebben, daar de bediening op dezelfde leest geschoeid is als dit bij de eerdere versies het geval was. Ik zou ze echter toch willen aanraden deze handleiding even door te lezen voordat ze het programma gaan gebruiken, dit om eventueel toch optredende problemen zo klein mogelijk te houden.

## Opstarten van DB3.00

Het programma wordt op de gebruikelijke manier geladen en daarna opgestart door middel van het linken naar #9000. Het geheugen gebied dat DB3.00 nodig heeft ziet U weergegeven in de memory map in figuur 1.Het eerste menu dat verschijnt na opstarten is weergegeven in figuur 2.




Figuur 1. Geheugen gebruik DB3.0X

We zien in bovenstaande figuur dat in tegenstelling tot de eerdere versies van DBASE het bestand nu vanaf #2902 wordt geladen zodat bij een <BREAK> niets meer verziekt wordt. De #XXXX waarde is natuurlijk afhankelijk van de hoeveelheid geheugen die een ieder tot zijn beschikking heeft.

```
                    MAIN-MENU

1..CREATE A NEW DATABASE              (C)
2..LOAD AN EXISTING DATABASE          (<)
3..EXECUTE DOS/COS COMMAND            (*)
4..DATAFILE DIRECTORY                 (.)
5..QUIT                               (Q)

:PLEASE ENTER COMMAND
```

Figuur 2. Hoofd menu.

Ik zal nu deze keuze mogelijkheden één voor één toelichten.

---

Het MAIN-MENU.

---

-CREATE.

Create gebruikt U om een nieuw databestand te definiëren. Dit is een zeer belangrijk onderdeel van het totale bestandsbeheer. U kunt namelijk een eenmaal gedefiniëerde opbouw NIET meer wijzigen en U moet dus goed bekijken welke velden U nodig denkt te hebben. Dit kunt U bijvoorbeeld doen door te bekijken op welke manieren, de informatie later gesorteerd dient te worden of welke gegevens als zoekcriteria moeten dienen.
Het definiëren van een bestandsmodel gaat op de volgende manier: U geeft allereerst het commando "C", er verschijnt dan een leeg datascherm, de cursor bevindt zich op de eerste positie van het zogenaamde titelveld. Hier kunt U de naam van het bestand invoeren bijv.: "MUZIEK ARCHIEF". Als U ter beeindiging van deze invoer op de returntoets drukt gaat de cursor naar regel 1 van het Dataveld. Hier kunt U de namen van de verschillende velden invoeren, met een maximum van 10 velden. De velden hebben ieder een maximale lengte van 23 karakters. Indien U met uw invoer over deze limiet heen probeert te komen wordt de invoer niet geaccepteerd en klinkt er een pieptoon ter indicatie. Na invoer van het laatste veld verschijnt het zogenaamde "WORK-MENU" op uw scherm, waarover straks meer.

-LOAD.

Deze keuze gebruikt U indien U een al eerder gedefiniëerd bestand van TAPE dan wel DISK wil laden. Het programma vraagt dan om de bijbehorende filenaam in te toetsen. (Er wordt overigens een controle op de lengte van de filenaam uitgevoerd, bij DOS maximaal 7 karakters en bij COS 13). Als het laden voltooid is verschijnt de tekst: "READY" en U kunt dmv. een toetsdruk verder gaan waarna het "WORK-MENU" verschijnt.

- OVERIGEN.

De overige 2 (bij COS) of 3 (bij DOS) commando's zijn meer algemeen. QUIT spreekt dacht ik voor zichzelf en dient ter beeindiging van het programma. Hierbij moet nog vermeld worden dat voor het verlaten van het programma het bestand gesaved dient te worden, tenminste als men dit wenst.
Het commando "." of "DATAFILE DIRECTORY", dat alleen in de DOS versie aanwezig is, laat alle files, die onder qualifier "D" op diskette staan, zien. Files die onder DB3.00-D gesaved worden, krijgen automatisch qualifier "D".
Het laatste commando uit het MAIN-MENU is het "*"-commando. Dit maakt het mogelijk om vanuit DB3.00 alle normaal aanwezige *-commando's uit te voeren. Het is natuurlijk ook mogelijk om eigen uitbreidingen die op ster commando's gebaseerd zijn vanuit DB3.00 uit te voeren, zolang deze zich niet op het A-blok bevinden.

---

## Het WORK-MENU

---

Als U, vanuit het Main-Menu, een bestand geladen of een nieuw bestand gedefiniëerd heeft, komt U terecht in het zogenaamde WORK-MENU, dat de commando's kent zoals aangegeven in figuur 3.

```
                  WORK-MENU

   1..INSERT          (I)   4..PRINT        (P)
   2..EDIT            (E)   5..LIST         (L)
   3..DELETE          (D)   6..SORT         (S)

   7..DEFINE MATCH RECORD                   (M)
   8..DEFINE PRINT LAYOUT                   (A)
   9..EXECUTE DOS/COS COMMAND               (*)
  10..DATAFILE DIRECTORY                    (.)
  11..SAVE DATAFILE                         (>)
  12..QUIT                                  (Q)

  :PLEASE ENTER COMMAND.
```

Figuur 3. Het WORK-MENU.

U ziet dat het menu in twee helften verdeeld is, nl de commando's 1 t/m 6 en 7 t/m 12. Het verschil tussen deze groepen zit in het feit dat de eerste zes commando's echt op afzonderlijk records werken en de rest meer algemene commando's zijn, die niets aan het bestand veranderen. Er wordt in DB3.00 niet meer met record nummers gewerkt, zoals dat in de eerdere versies het geval was. Hiermee is tevens de beperking van maximaal 253 records komen te vervallen. Het aantal records wordt nu alleen nog bepaald door de beschikbare hoeveelheid geheugen ruimte. U zult zich misschien afvragen hoe we nu bepalen welke records we een bepaalde bewerking willen laten ondergaan. Dit gebeurt mbv. het zogenaame MATCH-record. Hierin staan een aantal (tot 10) criteria waaraan de te behandelen records moeten voldoen. De commando's 2 t/m 5 worden alleen op DEZE records uitgevoerd. Als U bijv. alle boeken van een

bepaalde auteur uit wilt printen, zet U in het MATCH-record: AUTEUR=BERTUS AAFJES en daarna zal het commando PRINT alleen nog maar uitgevoerd worden op de records die aan dit criterium voldoen. Op dezelfde manier gaat U te werk als U bepaalde records wilt editen. Bijv. in een bepaalde woonplaats is bij alle telefoonnummers een "1" voor het abonnee nummer gekomen. U zet nu in het MATCH-record: PLAATS=BLARICUM en voert daarna de Edit opdracht uit, waarna U alle records voorgeschoteld krijgt waarin de plaatsnaam gelijk is aan BLARICUM. U kunt nu in deze records makkelijk even het telefoonnummer wijzigen. Het definiëren van het MATCH-record gaat overigens mbv. het commando "M".

-Define MATCH-record.

U krijgt een leeg record voorgeschoteld waarin U, net als bij het invoeren van een nieuw record, allerlei informatie kunt opslaan. Het verschil is dat deze informatie nu vergeleken gaat worden met de informatie uit de afzonderlijke records, die voorkomen in het bestand. De manier van vergelijken wordt bepaald door het eerste karakter van elke ingevoerde regel uit het MATCH-record. Hierbij heeft U 4 mogelijkheden en wel: ">", "<", "=" of een willekeurig ander karakter. De eerste drie mogelijkheden houden in dat bij het vergelijken de daarop volgende string met lengte n, groter, kleiner dan wel gelijk moet zijn aan de eerste n karakters van het te onderzoeken record. Hieronder dan enkele voorbeelden ter verduidelijking van het bovenstaande:

| Match string. | Record string | Resultaat. |
|---|---|---|
| >AUTO | BAND | TRUE |
| =AUTO | AUTOMOBIEL | TRUE |
| =AUTO | RACEAUTO | FALSE |

De vierde mogelijkheid houdt in dat de opgegeven string ergens in de te onderzoeken string moet voorkomen om een match te krijgen. Bijv:

| Match string. | Record string | Resultaat. |
|---|---|---|
| AUTO | RACEAUTO | TRUE |
| AUTO | AUTOMATISCH | TRUE |
| TOMAAT | AUTOMATISCH | FALSE |
| TOM.T | AUTOMATISCH | TRUE |
| ...O | AUTO | TRUE |

De laatste twee voorbeelden maken gebruik van de zogenaamde "wildcard" mogelijkheid. Een "." die in de match-string voorkomt, matcht met een willekeurig karakter uit de te onderzoeken string. Deze mogelijkheid mag natuurlijk ook bij de ">", "<", "=" zoekacties toegepast worden. De relatie tussen de 10 regels uit het MATCH-record is een EN-functie. Als dus aan één van de voorwaarden niet voldaan is zal de match, FALSE als resultaat opleveren. Indien U alle records wil selecteren, kunt U het matchrecord leeg definiëren. Een lege regel matcht met elke andere regel.

-De opties 10,11 en 12.

Deze commando's heb ik al toegelicht bij de beschrijving van het MAIN-MENU, ze kunnen vanuit het WORK-MENU op dezelfde manier gebruikt worden. Een verschil ligt bij het commando QUIT, waarbij niet het database programma wordt beeindigd, maar eem terugkeer naar het MAIN-MENU volgt.

-DEFINE PRINT LAYOUT.

Met behulp van dit commando kunt U bepalen hoe U de geselecteerde records op papier wilt hebben. Na geven van het commando "A" verschijnt het PRINT-MENU op uw scherm, met de volgende opties:

```
                PRINT-MENU.

    1......PRINTER SETTING      (S)
    2......DEFINE LAYOUT        (L)
    3......DEFINE HEADER        (H)
    4......LOAD DEFINITION      (<)
    5......SAVE DEFINITION      (>)
    6......PRINT                (P)
    7......DEFINE MATCH RECORD(M)
    8......QUIT                 (Q)

    :PLEASE ENTER COMMAND.
```

Figuur 4. Print-menu.

ad-1) Hiermee kunt U de gegevens van uw printer invoeren. Achtereenvolgens vraagt het programma U om de volgende gegevens in te voeren:
-Aantal kar./regel in NORMAL-MODE (Default=80).
-Aantal kar./regel in CONDENSED-MODE (Default=132).
-Aantal bruikbare regels per pagina. (Default=62).
-Totaal aantal regels per pagina (Default=72).
Indien U de default waarden wilt accepteren kunt U gewoon op <return> drukken waardoor de tussen haakjes vermelde waarde gehandhaafd zal blijven. Indien U een getal intoetst zult U merken dat de lengte van het getal maximaal 3 karakters bedraagt en dat het programma bij illegale invoer even piept om de gebruiker weer bij zijn positieven te brengen.

ad-2) Dit is dan het commando waarmee U de uiteindelijke vorm van uw geprinte overzicht vastlegt. Eerst volgt de vraag: "LABEL OR TABEL MODE? (T):". Tabel-mode staat voor een overzicht in een aantal kolommen naast elkaar, terwijl Label-mode gebruikt kan worden om bijvoorbeeld adres-stickers te printen. Indien U op bovenstaande vraag met "L" (Label-mode) antwoordt, moet U nog even opgeven hoeveel blanko regels U tussen elk label wilt hebben. Bij Tabel-mode volgt de vraag: "NUMBERING [Y/N] (Y):", waarmee U aangeeft of er voor elk afgedrukt record een volgnummer afgedrukt dient te worden. Dit nummer neemt 4 posities in, zodat een maximaal aantal items van 9999 kan worden afgedrukt, met nummering. Geen reële beperking

lijkt me.
Hierna volgt een aantal vragen per veld. Dus elke vraag moet 10x beantwoord worden, hetgeen redelijk snel kan doordat U bij het accepteren van de default waarden alleen op return hoeft te drukken. In onderstaande tabel heb ik geprobeerd de betekenis van elke vraag duidelijk te maken voor zowel Label- als ook Tabelmode.

| Vraag: | Default: | Tabel mode betekenis: | Label mode betekenis: |
|---|---|---|---|
| PRINT FIELD (Y/N) (Y): | YES | Een kolom met dit veld opnemen. | Een regel voor dit veld afdrukken. |
| PRINT HEADING (Y/N) (Y): | YES | Een kopje met de veldnaam boven de kolom opnemen. | Geen betekenis. |
| LEFT JUSTIFIED? (Y/N) (Y): | YES | De string wordt links aangesloten afgedrukt, NO=rechts. | Geen betekenis. |
| FIELD WIDTH (20): | 8 | De breedte van deze kolom. | Het aantal spaties dat voor de regel wordt afgedrukt. |

Het veld waarvan U momenteel de gegevens aan het invoeren bent kunt U herkennen aan de naam die in het titelveld wordt afgedrukt en het nummer dat hiervoor staat (0..9). Even een voorbeeldje van wat zoal mogelijk is. Stel U hebt een platen verzameling opgeslagen mbv DB3.00. Door de beperkte veldlengte hebt U uw recordstructuur als volgt opgebouwd:

```
+------------------------------+
:ARTIEST:PAUL SIMON &          :
:       :ART GARFUNKEL.        :
:TITEL  :THE CONCERT IN CENTRAL:
:       :PARK                  :
:                              :
:                              :
:                              :
:                              :
+------------------------------+
```

Op de printer wilt U hiervan een overzicht hebben dat er als volgt uitziet:

```
ARTIEST:                        TITEL:
----------------------------------------------------------------
PAUL SIMON & ART GARFUNKEL. THE CONCERT IN CENTRAL PARK
```

Hiertoe geeft U bij de vragen op:
-Veld printen: JA
-Kop printen: Veld 1:JA;
Veld 2:NEE;
Veld 3:JA;
Veld 4:NEE
-Links aansl.: JA
-Veldbreedte: Veld 1:0; Veld 2:32; Veld 3:0; Veld 4:32
Hierdoor worden dan de velden 1 en 2 achter elkaar afgedrukt en de velden 3 en 4 idemdito. Door eens flink te experimenteren zult U nog meer leuke mogelijkheden ontdekken.
Het programma bepaalt aan de hand van de totale breedte van alle kolommen samen of het geheel nog wel op het

papier past. Indien de breedte groter is dan de gedefiniëerde breedte in "NORMAL-MODE" zal de printer op condensed overgeschakeld worden. Indien de breedte zelfs het aantal karakters in "CONDENSED MODE" overschrijdt, wordt een waarschuwing gegeven bij uitvoering van het commando "P". U kunt dan nog afbreken en de definities aanpassen. U kunt echter ook gewoon laten printen. Het resultaat kan in dat geval wel een beetje tegenvallen.

ad-3) De HEADER slaat -in dit geval op de tekst die boven elke pagina afgedrukt wordt. U kunt linksboven de datum laten afdrukken (of een willekeurige andere string met een maximum lengte van 8 karakters). Rechts boven kan naar keuze een pagina nummer komen en midden boven een titel. Als U bij de vraag HEADING [Y/N]: met NO antwoordt, zal niets boven de pagina's afgedrukt worden, dus ook geen kop boven de kolommen. Alle invoer is weer op de gebruikelijke manier beveiligd tegen fouten. Er is voor elke invoer een maximale lengte en illegale karakters worden niet geaccepteerd.

ad-4) Hiermee kunt U een eerder ingevoerde print-layout definitie van TAPE dan wel DISK laden. In geval van disk zal deze onder qualifier "P" opgeslagen zijn. Zie ook "ad 5)".

ad-5) Zie "ad 4)" nu echter omgekeerd. De op dit moment in het geheugen aanwezige definitie wordt op TAPE dan wel DISK opgeslagen. Bij disk dus weer onder qualifier "P".

ad-6) Dit commando komt overeen met commando (4) uit het WORK-MENU. Het is hier toegevoegd om de mogelijkheid te geven om, zonder terugkeer naar het WORK-MENU, toch te kunnen printen. Alleen die records die voldoen aan de match criteria zullen worden afgedrukt. Er volgt nog een vraag of de uitvoer op de printer dan wel op het scherm dient te geschieden. Hiermee kunt U dus eerst kijken of de uitvoer aan uw verwachtingen voldoet, voordat er een daadwerkelijke uitvoer op papier volgt.

ad-7) Ook deze optie is alleen toegevoegd aan dit menu om de gebruikersvriendelijkheid zo hoog mogelijk te houden. We kunnen dus vanuit het PRINT-MENU, het MATCH-record wijzigen.

ad-8) En als laatste de mogelijkheid om weer naar het WORK-MENU terug te keren.

## -INSERT.

Het toevoegen van een nieuw record aan het bestand doet U met behulp van dit commando. U komt nu in de "FULL-SCREEN EDITOR" die nog uitvoeriger zal worden toegelicht bij de uitleg van het commando EDIT. Na invoeren van een record kunt U door middel van de toets "↑" terugkeren naar het WORK-MENU, of door een andere toets in te drukken het volgende record toevoegen aan het bestand.

## -PRINT en LIST.

Deze commando's laten U de inhoud van de verschillende records zien; LIST doet dit in standaard vorm op het scherm, terwijl

PRINT de records laat zien in een door uzelf gedefinieerde vorm.

-DELETE.

Alle records die matchen met het door U gedefiniëerde MATCH-record worden nu uit het bestand verwijderd. Daar dit aardig gevaarlijk is, wordt er eerst even om verificatie gevraagd, waarbij U een drietal mogelijkheden hebt, zoals weergegeven in figuur 5.
Het programma vermeldt eerst hoeveel records er voor verwijdering in aanmerking komen en geeft U de keuze tussen: 1)Gooi ze er maar uit of 2)Vraag per record of dit er wel uit mag. Voor het verwijderen is het commando "d" (Shift D) gekozen, om nog een extra beveiliging te geven tegen ongewenst verwijderen van data. De derde en laatste mogelijkheid spreekt voor zichzelf: Terug naar het hoofdmenu.

```
DELETE-MENU

MARKED ARE:20 RECORDS

1...DELETE ALL                (d)
2...SELECTIVE DELETION        (S)
3...QUIT                      (Q)

:PLEASE ENTER COMMAND.
```

Figuur 5. Delete-menu.

-SORT.

Hiermee kunnen we de records in het databestand in een bepaalde volgorde zetten. Het bestand wordt met behulp van "BUBBLESORT" gesorteerd op het veld dat we bij de gestelde vraag opgeven. Doordat bubblesort gebruikt wordt kunnen we bijvoorbeeld eerst op veld 1 sorteren en daarna op veld 2. Hierdoor is het bestand dan volgens veld 0 gesorteerd en ondergesorteerd naar veld 1.

-EDIT.

Het laatste commando tenslotte, is bedoeld om bestaande records te wijzigen. De records die aan de match criteria voldoen worden achtereenvolgens aan U aangeboden op het scherm, waar U ze met behulp van de al eerder genoemde, Full-screen editor onder handen kunt nemen. Wat houdt dat nu in, een "FULL-SCREEN EDITOR"? In het kort gezegd het record wordt net zo opgeslagen als het op het scherm staat. U kunt met de cursor toetsen ergens op scherm gaan staan en iets wijzigen, zonder dat U daartoe al het voorliggende met behulp van de copy toets hoeft te passeren. Bij het invoeren van karakters schuift de rest van de tekst naar rechts op. Er wordt dus echt tussengevoegd. Verder kunt U het karakter links van de cursor verwijderen mbv. de <DELETE> toets en met <CTRL-G> het karakter rechts van de cursor. Als het record de

door U gewenste vorm heeft verkregen, kunt U het edit proces afbreken door <CTRL-K> te toetsen, of door op de laatste regel van het record <RETURN> in te drukken. Het programma vraagt dan: "RECORD OK [Y/N]:", waarbij U, als U NO antwoordt weer het zelfde record mag editen, terwijl bij YES het volgende record volgt, of een terugkeer naar het WORK-MENU indien U juist het laatste record geedit heeft.

Hiermee zijn dan alle beschikbare commando's van DB3.00 besproken. Ik hoop dat U hiermee over voldoende informatie beschikt om prettig met het programma te kunnen werken.

Opm-1) Er zijn twee versies van DB3.00 namelijk: DB3.00-D en DB3.00-C. De eerste is bedoeld voor mensen die een diskoperating systeem draaiende hebben op hun machine, en de tweede vor de COS gebruikers.

Opm-2) Om het programma geschikt te maken voor zoveel mogelijk printers is er een tabel opgenomen waarin de besturingscodes voor de printer staan. De versie die in het archief komt, bevat de codes voor een EPSON RX80. Indien U in het bezit bent van een andere printer moet U deze tabel wijzigen.

| Adres: | Default inhoud: | Actie: |
|---|---|---|
| #AFF0 | #1B,#4F | Voer een eenmalige |
| #AFF2 | #00,#00 | initialisatie van |
| .. | ... | de printer uit. |
| #AFFA | #00. | MAXIMAAL 11 BYTES. |
| #AFFB | #00 | Karakter in #FE |
| #AFFC | #12,#00 | Cancel condensed. |
| #AFFE | #0F,#00 | Set condensed. |

## *Addendum bij DB3.00 --> DB3.01*

Bij gebruik van DB3.00 bleken in de praktijk nog een aantal punten voor verbetering vatbaar. Deze wijzigingen zal ik hier beschrijven.
Allereerst: in de COS versie was een DOS-call blijven staan, waardoor deze versie niet, of zeer onbetrouwbaar, draaide op machines zonder DOS-rom. Deze fout is nu opgeheven, verder zijn er enkele inkonsekwenties in de bediening en enkele taaltechnische fouten verholpen.

Beschrijving van de wijzigingen/verbeteringen:

-1) In het WORK-MENU ontbrak het "SAVE DATAFILE (>)" commando, dit is toegevoegd.
-2) Het stoppen van het uitprinten van gegevens met de <SHIFT> -toets is nu zowel in table- als in labelmode mogelijk (voorheen alleen in tablemode).

-3) De <LOCK>- en <COPY>-toets leverden bij het editen soms problemen op, deze zijn nu verholpen. Een gevolg hiervan is echter dat de toetsen niet meer werken in het Database programma, daar ze echter toch niet nodig zijn is dit geen probleem.

-4) Bij het afdrukken op het scherm wordt page-mode aangezet, waardoor we de uitvoer beter kunnen controleren. De printer initialisaties worden alleen naar het scherm gestuurd, waardoor problemen met deze codes (bv.$2) op het scherm vermeden worden. Verder wordt na afloop van het printen op een toetsdruk gewacht waardoor we ook de laatste pagina kunnen bekijken.

-5) De sorteer routine werkte niet goed, dit is verholpen.

Uit reacties van gebruikers is gebleken dat het geheugen gebruik van DB3 nog niet helemaal duidelijk is. Voor we DB3 kunnen runnen moeten we het volgende geheugen beschikbaar hebben:

- #2900..#XXXX voor opslag data.
- #8200..#9000 werkgebied DB3.XX
- #9000..#B000 programma.

Vooral het laatste punt wordt vaak over het hoofd gezien. We hebben hiervoor in de eerste plaats RAM van #9800..#A000 nodig (dus gestapeld hoog geheugen) en in de tweede plaats RAM op het A-blok !! Een bijkomend nadeel hiervan is dat mensen die met cassette werken hun FAST-COS routine kwijt zijn. Mogelijke oplossingen hiervoor zijn:

- Gebruik van P-Charme op #1000 (zie ook ACB 2.4/5)
- FCOS routine op bv #7000 (zie ook AN 4.6 pg.94)

Een Trucje:
-----------

Indien we per ongeluk op <BREAK> drukken en DB3 opnieuw opstarten met LINK#9000, kunnen we de data waarmee we bezig waren terug krijgen door een LOAD commando (= '<') te geven en hierin een fout te forceren. Dit kan bij DOS eenvoudig, nl door alleen <CR> in te toetsen op de vraag "FILENAME:" en bij COS door na de PLAY TAPE melding op <SPATIE> gevolgd door <CTRL> te drukken. Hierna volgt de melding: "error. PRESS A KEY", waarna we weer in het WORK-MENU terecht komen, waar al onze ingevoerde data weer beschikbaar is.

Opmerking:
----------

Als we met "MATCH", records hebben uitgekozen om te editen en we editen ze ook daadwerkelijk, behoren ze na het editen niet meer tot de verzameling van "gematchte" records, ook als we dit op grond van hun inhoud wel zouden verwachten. Dit is bewust zo geimplementeerd om te bereiken dat we elk record maar 1 keer voorgeschoteld krijgen en niet na het editen nog een keer.

Verantwoording:
---------------

Bovenstaande verbeteringen kwamen tot stand na zorgvuldig testen van DB3.00 door Ronald Boers, waarvoor dank.
Op- en aanmerkingen worden zeer op prijs gesteld door de schrijver dezes.

Heeft u ook wel eens dat er boven uw programma's een programma staat waar u niets meer mee kunt doen.
Laden gaat niet meer en het deleten er van gaat ook niet, erg vervelend want zo'n programma gebruikt toch Disk ruimte en het staat zeer rommelig in de CATALOG. Voorbeeld van zo'n Disk CATALOG:

```
*CAT
DISC-TITEL   DRIVE 0 QUAL    TEST2
      TEST1
 :     BLINK        CHECASS
       DISCSOF      DRASOFT
       DRSOFT      #DSOFT
```

In dit voorbeeld staan de twee bovenste programma's dus boven de eerste ":" en waren daar niet meer weg te krijgen.

De oplossing hiervoor:
======================

Als u een XDUMP maakt vanaf geheugen locatie #2000, zal de catalog informatie gedumpt worden.

Voorbeeld: XDUMP #2000,
==========

```
             D  I  S  C  -  T  I  T
      2000: 44 49 53 43 2D 54 49 54
             T  E  S  T  2
      2008: 54 45 53 54 32 20 20 00
             T  E  S  T  1
      2010: 54 45 53 54 31 20 20 0A
             B  L  I  N  K
      2018: 42 4C 49 4E 4B 20 20 20
                D  I  S  C  S  O  F
```

Het laatste byte op iedere regel is nu de QUALIFIER die bij het op die regel staande programma hoort. De DEFAULT QUALIFIER is #20 een spatie dus.
Bij de programma's TEST1 en TEST2 is op een of andere manier de QUALIFIER gezet op resp. #0A en #00.

Hoe kan dat nou?
================

Het blijkt dat de DOS ook nog een aantal bytes gebruikt uit het cassette operating systeem in de ZERO-PAGE.
De waarde van de qualifier wordt namelijk bewaard op geheugen-locatie #CD van de COS werkruimte.
Probeer dit maar eens door een BREAK te geven de DOS aan te zetten en daarna te kijken wat er op #CD staat, dit zal de default waarde van de qualifier zijn namelijk #20.
Toets nu in *SETA en dump opnieuw deze locatie. Deze is nu veranderd in #41 de ASCII code voor de letter A.
Zo kan het dus gebeuren dat door het cassette operating systeem of door een bepaald commando deze locatie wordt veranderd. Dit kan gebeuren in b.v. een schakelsoft systeem waar vaak tussen de DOS en de COS wordt geschakeld.
Een remedie zou hiervoor dus zijn om bij het overschakelen locatie #CD te laden met een spatie (#20).
Het kan dus zijn dat een aantal van uw programma's met een zeer vreemde qualifier zijn weggeschreven.

Hoe verhelpen we dit:
====================

We kijken naar het laatste byte in de regel met de programma-naam die we willen verwijderen of herstellen. Hier staat in ons voorbeeld #0A voor prgramma TEST1 en #00 voor programma TEST2.
We typen dan in:

```
        *SET(cntr j) (return)
        *DELETE TEST1
  en    *SET(cntr @) (return)
        *DELETE TEST2
```

en beide programma's zijn nu verwijderd.

Meestal zal het voorkomen dat zo'n programma een qualifier heeft meegekregen wat een CONTROL CODE waarde heeft zodat we achter het *SET commando dus een CONTROL functie moeten ingeven.

CONTROL WAARDEN:
================

```
   CNTR @ = 0      CNTR A = 1
   CNTR B = 2      CNTR C = 3
   CNTR D = 4      CNTR E = 5
   CNTR F = 6      CNTR G = 7
   CNTR H = 8      CNTR I = 9
   CNTR J = A      CNTR K = B
   CNTR L = C      CNTR M = D
   CNTR N = E      CNTR O = F
   CNTR P = 10     CNTR Q = 11
   CNTR R = 12     CNTR S = 13
   CNTR T = 14     CNTR U = 15
   CNTR V = 16     CNTR W = 17
   CNTR X = 18     CNTR Y = 19
   CNTR Z = 1A
```

Alle waarden zij in Hex.

Op deze manier kunnen we dus die "irritante programma's" waar we nooit wat mee konden doen opnieuw laden, deleten of terug schrijven met de goede qualifier.

Noot van de redactie:
In het bovenstaande verhaal wordt de zero-page problematiek met de DOS aangesneden. Zodoende wil ik nog enige informatie toevoegen. Voor DOS gebruikers zijn er meerdere zero-page adressen die perse NIET gebruikt moeten worden:
#AC: De huidige qualifier.
#B9: File-handler voor EXEC file.
#BA: File-handler voor *SPOOL file.
#C0: Geeft aan welke file's zijn geopend.
#C2: File-handler voor random file.
#CD: De in gebruik zijnde qualifier.
#EE: De aangewezen drive.

Het is niet zo dat de DOS zero-page adressen van de COS gebruikt, daar dit twee verschillende operating systems zijn, die niet gelijktijdig kunnen en mogen worden gebruikt.

# muvar

## MULTILETTERIGE VARIABELEN

Ik had net een programma ontwikkeld welke multiletterige variabelen toeliet zowel in assembler (symbolische assembler) als in basic toen Frans van Hoesel kwam met zijn MINIAS. In eerste instantie liet ik het hierna zitten, want zijn programma was immers al gepubliceerd. Een facet van mijn programma pakt minias echter nog niet aan en dat is de mogelijkheid om vanuit basic multiletterige variabelen te definieeren. Dit gedeelte heb ik dan ook uit mijn programma gesloopt met onderstaand resultaat.

De multiletterige variabelen kunnen alles zijn zolang zij uitsluitend de letters A t/m Z bevatten en dus geen kleine letters. Tevens mag de variabele niet beginnen met een bestaand statement. Het maximale aantal letters per variabele is 7 en er kunnen maximaal 30 variabelen worden gebruikt. Het is echter wel mogelijk om gelijktijdig de ATOM's eigen variabelen A t/m Z te gebruiken. Het programma pretendeerd niet perfect te zijn maar is slechts bedoeld om aan te tonen dat het mogelijk is. Het lijkt mij zinvol om het een en ander te combineren met minias.

In het onderstaand programma zijn regels 1070 en 1080 nodig als initialisatie. Dit kan echter ook automatisch met een boot up. Het programma zoals het hier staat kan nog niet samenwerken met p-charm maar dit is eenvoudig aan te passen door regel 1080 te veranderen in p-charm vectoren.

De rest van het programma laten het gebruik zien van deze variabelen. Tevens is het mogelijk om in basic variabelen te definieeren die dan in een assembler programma weer worden gebruikt als een vaste label waarde.

De volgende errors kunnen door het programma worden gegenereerd:

ERROR 49 TABEL VOL. Er zijn meer dan 30 variabelen in gebruik.

ERROR 50 De variabele bevat een element welke niet A t/m Z is
Het maximale aantal letters van 7 is overschreden.
Tijdens het definieren van een variabele is = niet gebruikt.

ERROR 51 Er is een fout geconstateerd tijdens de uitvoer van een functie. b.v. PLOT=90. Hier is de variabele een bestaand statement, hetgeen niet is toegestaan.

```
100DIM LL(50)
110F.I=0TO50;LL(I)=-1;N.
120P.$21
130F.I=0TO1
140P=#3700
150[
160:LL25JMP LL18
170PLA;PLA;CMP@94;BEQ LL7
180CMP@29;BEQ LL25
190PHA;PHA;JMP#C9D8
200:LL7PLA;CLD
210LDY#03;LDX@#00;STX#91;DEX
220DEY
230:LL1INX;INY
240:LL3LDA#9800,X;CMP(#05),Y;BEQ LL1
250CMP@#FF;BEQ LL4
260:LL2INX;LDA#9800,X;CMP@#FF;BNE LL2
270INC#91;INX
280LDA#9800,X;CMP@#FF;BEQ LL9
290LDY#03;JMP LL3
300:LL4DEY
310:LL16INY;LDA(#05),Y;CMP@#20;BEQ LL16
320CMP@#3D;BEQ LL12
330LDY#03;INC#91;JMP LL2
340:LL12STY#03
350LDA#91;CMP@#1E;BMI LL5
360LDA@#31;PHA;PHA;JMP#C9D8
370:LL5ASLA;ASLA;LDY#03;STA#91
```

```
380JSR#C4DE;LDY#91
390LDX#04
400LDA#15,X;STA#9900,Y
410LDA#24,X;STA#9901,Y
420LDA#33,X;STA#9902,Y
430LDA#42,X;STA#9903,Y
440DEC#04;JMP#C55B
450:LL9LDY#03;TXA;PHA
460:LL11LDA(#05),Y;CMP@#3D;BEQ LL10;CMP@#20;BEQ LL14
470CMP@#0D;BEQ LL13;CMP@CH";";BEQ LL13
480CMP@65;BMI LL13;CMP@90;BPL LL13
490STA#9800,X;INX;BEQ LL13
500INY;CPY@#09;BNE LL11
510:LL13LDY#03;PLA;TAX
520LDA@#FF;STA#9800,X;STA#9801,X;LDA@#32;PHA;PHA;JMP#C9D8
530:LL10PLA;LDA@#FF;STA#9800,X;STA#9801,X;JMP LL4
540:LL14INY;LDA(#05),Y;CMP@#3D;BEQ LL10;CMP@#0D;BEQ LL15
550CMP@CH";";BEQ LL15;BNE LL14
560:LL15JMP LL13
570:LL18PLA;CLD;TSX;INX;INX;TXS
580LDY#03;STY#90;LDX@#00;DEY;DEX
590LDA@#00;STA#91
600:LL19INX;INY
610:LL20LDA#9800,X;CMP(#05),Y;BEQ LL19;CMP@#FF;BEQ LL29
620:LL21INX;LDA#9800,X;CMP@#FF;BNE LL21;INC#91;INX
630LDA#9800,X;CMP@#FF;BEQ LL31;LDY#03;JMP LL20
640:LL31JMP LL24
650:LL29DEY
660:LL22INY;LDA(#05),Y;CMP@#20;BEQ LL22
670CMP@CH"+";BEQ LL26
680CMP@CH";";BEQ LL26
690CMP@#0D;BEQ LL26
700CMP@CH")";BEQ LL26
710CMP@CH"'";BEQ LL26
720CMP@CH",";BEQ LL26
730CMP@CH"%";BEQ LL26
740CMP@CH"&";BEQ LL26
750CMP@CH"*";BEQ LL26
760CMP@CH"-";BEQ LL26
770CMP@CH"/";BEQ LL26
780CMP@CH":";BEQ LL26
790CMP@CH"?";BEQ LL26
800CMP@CH"@";BEQ LL26
810CMP@CH"""";BEQ LL26
820CMP@CH"!";BEQ LL26
830CMP@CH"#";BEQ LL26
840CMP@CH"$";BEQ LL26
850CMP@CH"(";BEQ LL26
860LDY#90;STY#03;INC#91;JMP LL21
870:LL26STY#03;LDX#04;LDA#91;ASLA;ASLA;TAY
880:LL23LDA#9900,Y;STA#16,X
890LDA#9901,Y;STA#25,X
900LDA#9902,Y;STA#34,X
910LDA#9903,Y;STA#43,X
920INX;STX#04;LDY#03;LDA@#00;RTS
930:LL24LDA@#33;PHA;PHA;JMP#C9D8
940]
 950N.;P.$6;WAIT
 960$#9800="↑";!#9801=#FFFF
 970?#202=#03;?#203=#37
 980TEST=#12345678
 990P.&TEST'
1000P.&(TEST+1)'
1010OSCHRD=#FFF4
1020P=#2800;[
1030:LL45LDA@#42
1040JSR OSCHRD;RTS;]
1050LINK LL45
1060END
```

De kaart kan zowel in de kast als in rack geplaatst worden.
Bij plaatsing in rack wordt gebruik gemaakt van een 64-polige(haaks) male connector. Verbreek voor monatage eerts de verbindingen tussen de connectorrijen op de kaart.

Bij plaatsing in de kast, komt de kaart in de derde connector-rij , gerekend vanaf de achterkant van de ATOM-print.
Plaats hiertoe een rechte 64-polige female connector op de binnen connector van de ATOM-print.
Plaats op de kaart een rechte 32-polige (enkele rij) male-connector.
Indien de geheugenkaart al geplaatst is, wordt op de geheugen-kaart aan de componentenzijde een rechte enkelrijigen male-connector (32-polige) geplaatst, en op de schakelkaart een rechte 64-polige femaleconnector.
Bij plaatsing in de kast komt de connector aan de soldeer zijde van de schakelkaart. Bij gebruik in rack, komt de connector aan de componenten zijde.(dit is de zijde met de witte opdruk)

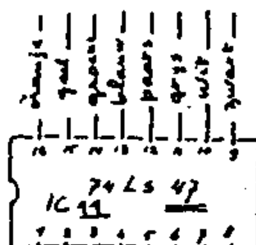
## DE ONDERDELEN VAN DE PRINT:

| | | | |
|---|---|---|---|
| R1-23 | 33 Kohm | IC1 | 74LS244 |
| R24 | 680 ohm | IC2 | 74LS245 |
| R25 | 33 Kohm | IC3 | 74LS244 |
| R26 | 470 ohm | IC4 | 74LS14 |
| R27 | 56 ohm | IC5 | 74LS138 |
| R28 | 220 ohm | IC6 | 74LS273 |
| R29 | 1 Kohm | IC7 | 74LS154 |
| R30 | 15 Kohm | IC8 | 74LS133(meegeleverd) |
| R31-32 | 33 Kohm | IC9 | 74LS10 |
| | | IC10 | 74LS32 |
| C1 | 33 uF (tantaal) | IC11 | 74LS47 |
| C2-4 | 15 uF (tantaal) | IC12-13 | CD4556B |
| C5-9 | 100 nF | IC14-19 | 2532 |
| C10 | 15 uF (tantaal) | IC20-27 | 6116,2016 |
| C11 | 100 nF | | |
| | | D1 | OA95 of 1N4001 |
| T1 | BC547B | D2 | rode led 3mm |
| | | D3 | 1N4001 |

INDIEN GEEN , OF NIET'OPLAADBARE ,BATTERIJEN WORDEN GEBRUIKT, DAN MOET WEERSTAND R24 WEGGELATEN WORDEN.

Bij uitlezing d.m.v. een display, is het volgende nodig :
IC11 wordt gebruikt om een cijferdisplay aan te sturen.
Wanneer het display niet wordt gebruikt, dan kan dit IC weggelaten worden.
Als aanvulling op de onderdelenlijst bij gebruik van een display komen dan nog :
1 x 7-segments display TIL 312
7 x weerstand 470 ohm

Het display wordt dan op de volgende manier aangesloten.






## ALGEMENE GEGEVENS :

De schakelkaart voegt een tiental 4K.sockets toe aan de 'Hardware' van de ACORN ATOM.

De situering in de MEMORY-map van deze sockets is :

6 x een 4K. socket vrij instelbaar
4 x een 2K. socket vrij instelbaar

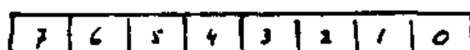
De Sockets 20 t/m 27 zijn geschakeld voor bezetting met CMOS geheugen IC's. één, of gestapeld twee per socket dus resp. 2K. danwel 4K. per socket.

De sockets 14 t/m 19 zijn geschakeld voor bezetting met Eprom's van het type 2532.
Ten behoeve van de CMOS-geheugens bevat de kaart de BATTERY-BACKUP schakeling, alsmede de aansluiting voor een WRITE-PROTECT schakelaar.

Toepassing van de kaart op gebieden die oorspronkelijk niet buiten de bus zitten verlangt aanpassing van het bus-enable signaal. zie hiervoor de tekening.
De kaart zelf is geheel gebufferd voor alle adressen en datalijnen. De data-buffer is geklokt.
Ten behoeve van de schakel-doeleinden wordt tevens adres XFFF buiten de buffer gebracht. Het schakelen geschied hiermede geheel software matig.

De kaart bevat een socket om een 74LS47 in te plaatsen, dit voor het aansturen van een 7-segments display.
Bij het aanzetten van de computer RESET de schakeling naar de laagtste geschakelde voet.

Adresseren met schakelbyte #BFFF
(Write only, kan niet teruggelezen)



IC's 20,22,24,26 komen in de sockets op de kaart.
IC's 21,32,25,27 komen hier gestapeld op te zitten.

IC 20 A000 BLOK 1
IC 22 E000 BLOK 0
IC 24 A000 BLOK 0
IC226 E000 BLOK 1

Aaansluitpunten voor de gestapelde IC's, deze punten zitten tussen IC12 en IC13.

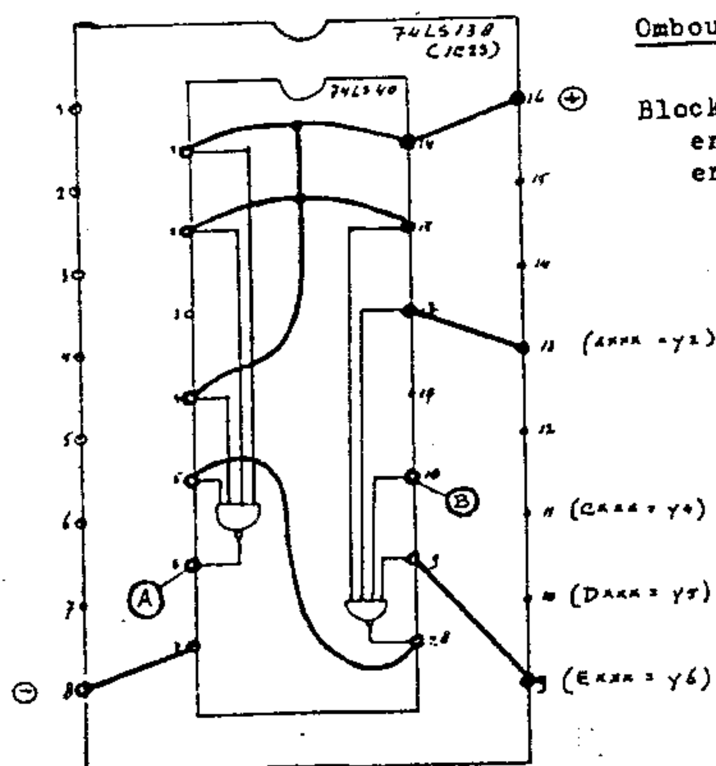
IC21 punt 1 A800 BLOK 0
IC23 punt 2 E800 BLOK 1
IC25 punt 3 A800 BLOK 1
IC27 punt 4 E800 BLOK 0

verder komen IC's 14 - 19 overeen met :

IC 16 AXXX blok 2
IC 17 AXXX blok 3
IC 18 AXXX blok 4
IC 19 AXXX blok 5
IC 14 AXXX blok 6
IC 15 AXXX blok 7

Hierbij is uitgegaan dat de punten A,B,E bij de 74LS154 op deze adressen zijn ingesteld.

draadb. op de print is een draadbrug om een data lijn door te geven aan 74LS47.




Ombouw IC 23

Block BFFF
en AXXX
en EXXX

'buiten de bus' brengen

Dikke rechte lijnen : te verbinden pootjes(4 stuks)
Dikke kromme lijnen : draadbruggen

A verbinden met IC 8 poot 4 (uit de socket buigen)

B verbinden met IC 49 poot 7 (terug in de socket)

N.B. : Van IC 23 alle poten terug in de socket.

# PROGRAM FLOW CONTROLLER

## VOORWOORD

De in deze handleiding gepresenteerde debugger zal niet veel weg hebben van de al bestaande debuggers voor de ATOM. Met deze debugger hoop ik ook een geheel nieuw concept te introduceren. Zo op het eerste gezicht lijkt deze debugger het midden te houden tussen een machinetaal monitor en een volwaardige debugger. Hoewel het wel degelijk mogelijk is deze debugger te 'misbruiken' als monitor, is zij er in de eerste plaats niet op gebouwd. Het executeren van een routine zal meestal beëindigd worden met een foutmelding, en dat is nou niet bepaald elegant voor een monitor. Verder worden de routines dermate vertraagd, dat het eigenlijk onzin wordt, PFC als een monitor te beschouwen.
Zoals al gezegd, er is gekozen voor een nieuw concept. Het is een beetje wennen, maar het biedt als zodanig vele voordelen. Het te bewerken programma is tot in de kleinste details beheersbaar.
Dat opent tevens naast debuggen een tweede perspectief. PFC is uitermate geschikt door zijn heldere representatie voor onderwijskundige doeleinden. Ik hoop dan ook dat ik PFC bij één van de vele cursusjes voor ATOM-bezitters weer terug zal zien.

## INLEIDING

De PROGRAM FLOW CONTROLLER, afgekort met PFC, is een veelzijdig hulpmiddel bij het debuggen van 6502-machinetaal programma's. Debuggen is een proces van testen en opsporen van fouten in programma's. Bij het debuggen komen vaak lange saaie stukken voor, delen van programma's die al getest zijn, maar ten behoeve van andere routines toch nog geëxecuteerd dienen te worden. Hierbij valt te denken aan delen van de ATOM interpreter en operating system, of aan eigen subroutines. PFC biedt de gebruiker faciliteiten om deze lange teststukken zo elegant en efficient mogelijk te passeren. PFC is daarvoor met een aantal unieke opties en commando's uitgerust. Om een mogelijke fout zo snel mogelijk te herkennen is PFC uitgerust met een aantal error en warning faciliteiten. PFC draagt er persoonlijk zorg voor dat executies een reële betekenis hebben. Hij controleert daarvoor uitgebreid de stack, subroutine levels en opcodes, en waarschuwt de gebruiker als er iets fout dreigt te gaan.
De efficiëncy van de debugger hangt echter in grote mate af van de manier waarop zij gebruikt wordt. Een goed testplan is daar-

bij onontbeerlijk. Er kunnen situaties in opgezet worden, controles uitgevoerd worden en programmalopen bekeken worden. PFC kan een handig hulpmiddel zijn bij het ontwikkelen van al uw programmatuur.

---

## BENODIGDHEDEN

---

PFC maakt gebruik van de VIA 6522 en werkt op interrupt-basis. Link LK2 moet in de ATOM gesloten zijn. Verder moet de floatingpoint-rom aanwezig zijn. PFC draait in zijn basisversie onder P-CHARME. Zij is echter ook als toolkit voor het A-blok verkrijgbaar. Alle programmatuur die gebruik maakt van interrupt-verwerking, dient uitgeschakeld te worden.

---

## GEHEUGENGEBRUIK

---

Afhankelijk van de versie die U heeft, heeft PFC zelf zijn plaatsje nodig in de ATOM. Daarnaast verbruikt PFC de volgende geheugen-gebieden:

- VIA 6522
- #2800-#2824
- #F,#E0,#0,#4,#3
- #64-#6B
- BASIC-variabelen P,X,Y,A,B,D,C,T,S,F
- BRK & IRQ vectoren

Let op: alle floatingpoint variabelen zijn nu bezet !
Access op deze gebieden kan de werking van PFC ernstig schaden. Denk hierbij vooral aan die routines in de ATOM die de VIA bewerken. Te denken valt hierbij aan de printer die aan de VIA kan hangen. Cntrl B zet niet alleen de printer aan, maar ook PFC uit !
De P-CHARME versie neemt plaats vanaf #8200 tot en met #9271.
De A-blok versie uiteraard vanaf #A000 tot en met #AFFF.

---

## MENGBAARHEID MET BASIC

---

PFC is ontworpen om in samenwerking met de ATOM basic en P-CHARME te kunnen werken. De statements werken zowel in direct mode als vanuit een BASIC-programma. Zie bijvoorbeeld ook het demo-programmaatje. Een aantal statements moeten echter door ESC afgesloten worden.
De 6502 registers zijn bereikbaar via de BASIC-variabelen. De volgende BASIC-variabelen zijn in gebruik:

bij onontbeerlijk. Er kunnen situaties in opgezet worden, controles uitgevoerd worden en programmalopen bekeken worden. PFC kan een handig hulpmiddel zijn bij het ontwikkelen van al uw programmatuur.

---

## BENODIGDHEDEN

---

PFC maakt gebruik van de VIA 6522 en werkt op interrupt-basis. Link LK2 moet in de ATOM gesloten zijn. Verder moet de floatingpoint-rom aanwezig zijn. PFC draait in zijn basisversie onder P-CHARME. Zij is echter ook als toolkit voor het A-blok verkrijgbaar. Alle programmatuur die gebruik maakt van interrupt-verwerking, dient uitgeschakeld te worden.

---

## GEHEUGENGEBRUIK

---

Afhankelijk van de versie die U heeft, heeft PFC zelf zijn plaatsje nodig in de ATOM. Daarnaast verbruikt PFC de volgende geheugen-gebieden:

- VIA 6522
- #2800-#2824
- #F,#E0,#0,#4,#3
- #64-#6B
- BASIC-variabelen P,X,Y,A,B,D,C,T,S,F
- BRK & IRQ vectoren

Let op: alle floatingpoint variabelen zijn nu bezet !
Access op deze gebieden kan de werking van PFC ernstig schaden. Denk hierbij vooral aan die routines in de ATOM die de VIA bewerken. Te denken valt hierbij aan de printer die aan de VIA kan hangen. Cntrl B zet niet alleen de printer aan, maar ook PFC uit !
De P-CHARME versie neemt plaats vanaf #8200 tot en met #9271.
De A-blok versie uiteraard vanaf #A000 tot en met #AFFF.

---

## MENGBAARHEID MET BASIC

---

PFC is ontworpen om in samenwerking met de ATOM basic en P-CHARME te kunnen werken. De statements werken zowel in direct mode als vanuit een BASIC-programma. Zie bijvoorbeeld ook het demo-programmaatje. Een aantal statements moeten echter door ESC afgesloten worden.
De 6502 registers zijn bereikbaar via de BASIC-variabelen. De volgende BASIC-variabelen zijn in gebruik:

- A accumulator
- X x register
- Y y register
- S stackpointer
- F status register, vlaggen
- P program counter

- T PFC register tijdsregistratie
- C PFC register count registratie
- D PFC register teller subroutine diepte
- B PFC register teller brk-routine diepte

De inhouden van deze variabelen zijn door de gebruiker te veranderen om zo tessituaties uit te werken. Het commando CLEAN cleared de PFC registers T,C,D,B.

---

## OPTIES

---

Om het testproces zo plezierig mogelijk te laten verlopen, zijn een 3 tal opties aangebracht. Deze 3 opties bepalen het karakter van de debugger in sterke mate. De 3 zijn als volgt:

- M Met de Monitor optie is het beeld te beïnvloeden. Als de debugger niet in de M optie staat ( M van monitoring ) , dan wordt er in het algemeen geen beeld geproduceerd. Dit geldt echter niet voor het TRACE commando, en wanneer er een ERROR conditie optreedt.

- S Met de Systeem optie wordt het gedrag bepaald wanneer de program counter >=#C000 is. Met de S optie aan zal de debugger dan net zolang doorwerken totdat zij weer uit het ATOM systeem gebied is. PFC beschouwt het systeem gebied als het ware als 1 instructie. PFC is altijd te stoppen door op de ESC toets te drukken.

- P Met de P optie ( van Procedure ) is het zelfde te bereiken, als met de S optie, maar dan nu voor subroutines. Wat PFC als een subroutine beschouwt hangt af van de waarden in de B en D BASIC-variabelen. Zijn deze groter dan 0 dan beschouwt PFC alles wat hierna komt als een subroutine.

Door deze 3 opties zorgvuldig te kiezen kan het te debuggen proces sterk versneld worden. Pas wel op: de ATOM tekstroutine #F7D1 eindigt niet met RTS, maar met PLA PLA, en dat wordt niet als het beëindigen van de subroutine herkend. De S optie is hier dus aan te bevelen.

## BEELDOPBOUW

Het scherm is verdeeld in 3 delen. De bovenste 3 regels zijn gereserveerd voor de ERROR en WARNING meldingen. Bij iedere executie worden deze 3 regels gecleared. De cursor wordt altijd buiten dit gebied gehouden.
De onderste 3 regels zijn gereserveerd voor de gebruiker. Hier kunnen de commando's worden ingetypt. Het middelste gedeelte wordt in de M optie gebruikt voor het display. Zie hiervoor ook het hoofdstuk 'DISPLAY INHOUD'. Iedere keer dat het beeld wordt opgebouwd, wordt de cursor ook buiten dit gebied gehouden. Zo worden conflicten tussen de gebruiker en PFC voorkomen. Cursor home betekent dus links bovenaan van de onderste 3 regels.

## VOORBEELD SCHERM

```
execution interrupted


    PROGRAM FLOW CONTROLLER


TIME:       0031   COUNT:      0011
SUB-DEPTH:  02     BRK-DEPTH:  00
OPTIONS:    M

  PC   A  X  Y     SP  NV BDIZC
FF02 38 F6 6A      F0  00101001
OPC.:F0 34         BEQ #FF38
>TIME 5
>
>
```

## DISPLAY INHOUD

In het display wordt alle informatie over de stand van zaken op overzichtelijke wijze gepresenteerd. De inhouden van de 6502 registers, alsmede de onderhanden zijnde opcode worden onderaan gedisplayed. Niet bestaande opcodes worden als instructie van lengte 1 afgedrukt.De disassembly verraadt dat het geen geldige opcode betreft. Instructies van lengte 3 worden eerst met het high-order gedeelte, dan pas met het low-order gedeelte gedisplayed. Dit is gedaan om de leesbaarheid te vergroten.
Bovenin staan de waarden van de 2 diepte tellers. Subdepth geeft het aantal subroutine niveau's aan, brkdepth het aantal

brk niveau's. Verder is nog in beeld de waarde van de tijd in microsecunden, en het aantal opcodes die geexecuteerd zijn.
Het beeld kan opgebouwd worden met de commando's CLEAN en VIEW.

---

ERROR'S & WARNINGS

---

PFC test op een aantal zaken om de kans een fout op te merken zo groot mogelijk te houden.
Er zijn 2 warnings:

- De BRK-WARNING waarschuwt als er een BRK geëxecuteerd is. Dit is meer een melding voor de gebruiker, dan een fout conditie.
- De interruptflag-warning. Als de interruptvlag van de 6502 geset is, voorkomt zij interrupts door de VIA. Om dit te voorkomen moet de vlag altijd gecleared zijn. PFC draagt hiervoor zelf zorg.

De WARNING dient om de gebruiker er op te wijzen dat het status register niet de correcte waarde bevat. Blijven nu nog de errors over.
PFC checkt de stack. De stackpointer moet zich bevinden binnen de volgende grenzen: #80<=S<=#E0 . Als hier niet aan is voldaan volgt een stack underflow/overflow ERROR.
Verder worden de RTS, BRK, RTI en JSR instructies bijgehouden. Een RTS teveel veroorzaakt een error. Informatie hierover staat in de 2 diepte tellers.
Verder wordt de onderhanden zijnde opcode nog getest. Is het geen geldige opcode, dan wordt de executie met een error melding tegengehouden. 65C02 Bezitters hebben dus pech.
Tevens veroorzaakt het indrukken van de ESC toets een error. Hiermee is de executie altijd te onderbreken.
Nadat er een error is vastgesteld stopt de debugger ALTIJD !

---

BRK HANDLING

---

Het is door PFC de gebruiker toegestaan om BRK's te executeren. Hierbij geldt wel als voorwaarde dat de BRK en IRQ vectoren niet vanuit het te debuggen machinetaal gedeelte geïnitialiseerd mogen worden. Het is wel toegestaan dit vanaf het toetsenbord in direct-mode te doen. Dit kan bijvoorbeeld met het commando BRK, maar ook met ?#204=#?? en ?#205=#?? .Omdat de ATOM de BRK vector steeds weer op #C9D8 zet ,is er voor gekozen de BRK te laten lopen via de IRQ vector op #204 / #205. Dit is de normale gang van zaken bij andere 6502 machines dan de ATOM. De uitvoering van de BRK kost 36 microseconden, en 10 geëxecuteerde opcodes. PFC gaat namelijk pas weer verder vanaf de vector op #204, en het aanloopgedeelte in de ATOM monitor wordt altijd ineens uitgevoerd. Hoewel de BRK een 1 byte instructie is, wordt hij tijdens executie opgevat als een 2 byte instructie. De program-counter wordt dus ook verhoogd met 2.

Ook dit is de normale gang van zaken bij 6502 machines. Het hier genoemde effect wordt het remeffect, ofwel de remweg van de BRK, genoemd.

---

## FLYBACK SIMULATIE

---

Om timing problemen te voorkomen wordt de flyback bij #FE66 en #FE6B gesimuleerd. Dit houdt twee zaken in. Ten eerste wordt er bij de simulatie geen rekening gehouden met het al of niet ingedrukt zijn van de RPT toets. Er wordt in alle gevallen gedaan alsof deze niet ingedrukt is. Normaal is dat geen probleem. Het is maar dat U het weet !
Ten tweede lopen alle systeemroutines die gebruik maken van de flyback nu veel sneller.

---

## MULTI-ATOM

---

Helaas is de ATOM niet geschikt voor het laten draaien van meerdere programma's tegelijk. En dat is nou wat wij met een debugger wel plegen te doen. Concreet houdt dat in, dat als je een routine gaat TRACEN die ook door PFC gebruikt wordt, het resultaat onvoorspelbaar is. Hierop is een grote uitzondering gemaakt ! Alle outputroutines zijn wel geschikt gemaakt voor multi-programma's. Zij kunnen dus wel probleemloos door PFC gevolgd worden. Dit is ook de oorzaak van het vrij grote geheugengebruik van PFC. De werkgebieden van de outputroutines worden steeds weer gesaved.

---

## DE COMMANDO'S

---

Nu het belangrijkste stuk van deze uitleg van de debugger. Bij alle statements moet in acht genomen worden dat subroutines in P optie, en binnen het ATOM systeem gebied in de S optie, als 1 instructie wordt opgevat. In het onderstaande wordt met expressie een expressie in de ruimste zin van het woord bedoeld. #202, A, A+3 bijvoorbeeld. In alfabetische volgorde volgen hier de statements die PFC rijk is:

- ↑ Met dit statement voert PFC 1 executie uit.Afhankelijk van de opties P & S kunnen dit meerdere instructies zijn.
- BRK Het BRK-statement initialiseert de BRK-vector , die geplaatst is op adres #204 / #205. Het is niet toegestaan om dit via het te debuggen programma te doen. Om dat te benadrukken is dit commamdo aan PFC toegevoegd.
  Syntax: BRK [expressie]

- CLEAN wist de diepte tellers, de tijd en het aantal opcodes. Daarnaast drukt zij ook het scherm 1 keer af.
- COND is wel de krachtigste opdracht aan PFC. Met deze opdracht is het mogelijk PFC te laten doorstomen totdat aan een bepaalde conditie is voldaan.
  Syntax: COND [register] [expressie]
  Voorbeëld:
  COND X 0 Laat PFC doorwerken totdat het X-register 0 is geworden. Voor register kan A,X,Y,P,S,F ingevuld worden. Dit statement is erg handig als er kleine loopjes doorlopen moeten worden. Stomvervelend werk bij andere debuggers. Toegevoegd zijn nog de volgende 2 extra commando's voor de register waarde. Een > teken laat PFC doorstomen totdat de program counter >= aan de waarde van de expressie is. Een M teken laat PFC doorstomen totdat het adres (waarde van de expressie) van waarde veranderd. Als je nu wilt weten waar een te debuggen programma zero-page-locatie #02 van waarde doet veranderen, dan is:
  COND M #02 voldoende.
- COUNT De syntax is: COUNT [expressie]. Dit statement zorgt ervoor dat er zoveel instructies worden afgewerkt als de exressie oplevert. COUNT 5 werkt 5 instructies af. De waarde moet liggen binnen 0 en 65535.
- DEBUG is een commando dat alleen de P-CHARME versie kent. Dit commando zorgt ervoor dat de tabel met statements aan P-CHARME wordt doorgelinkt. Andere al aangesloten tabellen blijven werken.Verder voert DEBUG het statement CLEAN uit, en zet het de registers op 0, de program counter default op #2000, en de vlaggen gecleared.
- DISAS Met dit statement kan een disassembly van het geheugen opgevraagd worden. Handig om te weten wat die opcodes nou ook alweer betekenden. DISAS wordt beëindigd met een ESC.
  Syntax: DISAS [expressie]
- HELP:HELP drukt een schermpje af waarin informatie staat over het gebruik van PFC.
- HEX Dit statement is alleen beschikbaar voor de A-blok versie. HEX laat de inhouden van de adressen vanaf het opgegeven adres zien. Het vervangt het HDUMP statement van de JOSBOX. HEX is alleen te stoppen met de ESC toets.
  Syntax: HEX [expressie]
- MEMORY Met dit statement kan het geheugen voor testdoeleinden snel gevuld worden. Syntax: MEMORY [expressie] . Vanaf het adres van expressie laat PFC zien welke inhouden het geheugen heeft. Telkens moet een nieuw byte ingetypt worden. Het intypen kan gestopt worden met een return.
- MOFF zet M optie uit.
- MON zet M optie aan.
- POFF zet P optie uit.
- PON zet P optie aan.
- SOFF zet S optie uit.
- SON zet S optie aan.
- STACK laat de stack vanaf stackpointer tot en met #01E0 zien.
- TIME doet hetzelfde als COUNT, maar dan met de tijd.
- TRACE Syntax: TRACE [expressie]
  Voorbeeld TRACE 4, single stept het programma met als wachttijd 4*1/4 seconde. Deze routine is alleen maar te stoppen met de ESC toets.
- VIEW laat het scherm zien, dat op dat moment correspondeert

met de registers van PFC.

Alle commando's beëindigen wanneer op ESC gedrukt wordt of wanneer er een ERROR-conditie optreedt. De BASIC-registers P, A, X, Y, S, F moeten natuurlijk gevuld worden.

---

## VOORBEELD

---

Het onderstaande voorbeeld geeft een demonstratie van PFC:

```
DEBUG
P=#2900; [; JSR #F7D1; ]
$P="HALLO BESTE MENSEN "; P=P+LENP; [; NOP; JMP #2900; ]
P=#2900
HELP
↑
↑
SON; ↑
STACK
VIEW
SOFF
TRACE 8
              <<<<<ESC.
COUNT 100
COND P #2900
```

---

## PROBLEMEN TIJDENS EXECUTIE

---

Als PFC het ineens laat afweten is er sprake van verboden access op het werkgeheugen van de debugger. Er is een CNTRL B naar de printer gestuurd, of de VIA of de BRK en IRQ vectoren zijn veranderd. Het kan ook zijn dat een systeem routine verkeerd wordt afgewikkeld ddor de debugger, omdat zij elkaar gewoon in de weg zitten.
Hier is echter niet veel aan te doen. Alle debuggers hebben hier in meerdere of mindere mate last van. PFC is uitgerust met wat hulpmiddelen die het debuggen van de testroutines wel mogelijk maakt. Dat heeft tot effect dat de cursor na bijvoorbeeld STACK op de oude plaats terug wordt gezet.
Er kunnen nog wat fouten in PFC zitten, maar PFC is zo uitgebreid getest, dat wat er nog aan problemen overblijft alleen nog maar het gevolg kan zijn van de 'shared facilities' ( shared = gedeeld ! ) .

PFC heeft er zoals in het vorige ACORNTJESBROOD al was aangekondigd er een broertje bij. Het betreft een speciale versie die geschikt is voor het parallel debuggen van andere toolkits. Dat dit niet zomaar gaat mag wel duidelijk wezen, zodat een uitleg toch op zijn plaats is.
Deze versie klapt zichzelf uit naar #8200- #9124, en schakelt daarna automatisch de te bekijken toolkit in. Daarna zal altijd weer de controle teruggegeven worden aan PFC.
PFC heeft daarbij het schakelnummer van de te debuggen eprom nodig, en het eprom nummer van zichzelf:
BASIC variabele N = te debuggen eprom
BASIC variabele V = PFC eprom
Initialiseer deze twee eerst voordat U wat voor commando dan ook uit PFC aanroept, anders zijn onverwachte effecten en vastlopen het gevolg.
Voorbeeld: voorheen JOSBOX ?#BFFF=6 en PFC ?#BFFF=7
Nu: N=6;V=7
Er wordt automatisch geschakeld in de bytes #BFFF en #B800. Eventuele schaduw adressen worden verwaarloosd !
Dat laatste is echter geen probleem want PFC keert altijd terug naar de schakelwaarde in Variabele V, zelfs als er een fout optreedt tijdens executie. Dat brengt mij tevens tot een tweede punt. In de handleiding wordt niet gesproken over eventuele ERROR NUMMERS. Dat is bewust gedaan omdat eventuele fouten of te maken hebben met de invoer, in welk geval de veroorzaker duidelijk is, ofwel met een foutieve toestand binnen PFC zelf. Deze laatste zijn echter niet zo duidelijk voor de gebruiker en altijd het gevolg van verboden access op eigen werkgeheugen !
Bovendien hebben de P-CHARME versie als de #8200 versie verschillende error nummers.
Bij de Toolkit versie is er sprake van een integrale ERROR 81 ongeacht de veroorzaker.
De toolkit versie gebruikt ook iets meer geheugen dan de andere versies:
#8200 - #9124
#A000 - #B000
#2800 - #2826
#66, #67, #68, #F, #0, #E0, #4, #69, #A
BASIC VARIABELEN: P X Y A S F B D T C N V

Blijft nog een opmerking over de ingebouwde checksums.
Bij alle 3 de versies is het zo dat:
A=0;FOR I=#XXXX TO #XXXX;A=A:?I;NEXT I;P.A
de waarde 0 moet opleveren.
Echter de P-CHARME versie verandert zichzelf ! Schrikt U niet, er is niets aan de hand, geen viezigheid hoor ! Deze versie werkt met een tabel die aan P-CHARME wordt doorgelinkt met het commamdo DEBUG. Het extensionbyte van deze tabel krijgt dan ipv. de waarde #FF een andere waarde. Zie ook P-CHARME handleiding.
Voor het checksum berekenen van deze versie moet dus eerst #8BFF op #FF gezet worden,waarna van #8300 tot #9272 gesommmeerd kan worden. Denk er dus aan dan U de checksum bepaalt voordat PFC gerunt wordt.

---

## REACTIES

---

PFC pleegt zo elegant mogelijk zijn gebruiker te bedienen en met zijn omgeving om te gaan. Het kan natuurlijk zijn dat er nog meer leuke statements bijgemaakt kunnen worden. Wat dat betreft ben ik nog lang niet uitgeput. Voorstellen, op- of aanmerkingen, evenals vermeende fouten in PFC zelf, zou ik gaarne willen vernemen. Het kan altijd nog beter !
Correspondentie adres:

Calslaan 52-13
ENSCHEDE
TEL: 053-893923

Ik wens U veel programmeerplezier toe en hoop dat PFC Uw inzichten in andermans programma's en de 6502 zal vergroten.
Als er eventueel wensen zijn voor PFC op andere plaatsen in de ATOM, dan wil ik dat wel voor elkaar maken. Probeer het niet zelf, er zitten nogal veel tabellen in PFC.
Tot slot: een aantal mensen zal PFC natuurlijk willen gebruiken om PFC te bekijken. Hierover twee opmerkingen. Ten eerste is er niets zo shared met iets anders als dat andere weer zichzelf is. Ten tweede is PFC niet optimaal geprogrammeerd. Ik zeg het maar alvast, daar hoeft U dus niet naar te zoeken !
Er is een versie in ontwikkeling die geschikt is voor het testen van andere eproms !

# SPEECH SYNTHESIZER

In ACORN NIEUWS nr. 1 van jaargang 3 (en ook in dit nummer red.) heeft Frank Cuypers een verhaal geschreven over de mogelijkheid van speech synthesis op de ATOM. Hij concludeerd dat fonemengenerators de meest zinvolle weg bieden naar computer spraak. Dit omdat zij het eenvoudigst zijn te programmeren en omdat zij slechts ongeveer 8 bytes per seconde spraak nodig hebben.
Er zijn meerdere fonemengenerators op de markt waarvan de meest bekende de SC-01 van VOTRAX is. In diverse tijdschriften hebben artikelen gestaan betreffende dit IC. Men liet zien dat het zelfs mogelijk is om de pitch (lees klankhoogte) alsmede de klokfrequentie te veranderen waarmee men het IC zelfs kan laten zingen. (dit is echter niet mogelijk met de SP 0256 AL2 die dadelijk zal worden beschreven ) Een ander IC die tevens een fonemengenerator bevat is de SP 0256 AL2 een IC die door William Stuart Systems LTD als SX-01 wordt aangeduid (hetzelfde IC dus). Dit laatste IC heeft een aantal vordelen boven de SC-01:

1) Er is geen +12V voeding voor nodig.
2) Het IC is belangrijk goedkoper dan de SC-01.
   (+/- fl 50.00 tegenover fl 200.00 voor de SC-01)

Daar ik slechts een zeer sumiere DATASHEET in mijn bezit heb weet ik niet precies waar de datalijnen A7 en A8 voor dienen. Het verbinden van A7 en A8 met +5V lijkt niets zinvols op te leveren maar als men deze lijnen aan massa legt dan verkrijgt men de beloofde fonemen. Een vergelijking tussen de beide IC's betreffende de kwaliteit van de gesproken text kan ik eveneens niet geven.
De SP 0256 AL2 is een 1 chip speech synthesizer die digitale filters, control rom en een pulse breedte modulator bevat voor een extern gefilterde analoge output. Het IC is voorgeprogrammeerd met een complete set fonemen of spraak componenten die opgeroepen en geactiveerd kunnen worden door de code getallen (0-63) te laden via de 8 data lijnen. Alle in en outputs zijn LS TTL compatibel en het geheel vergt slechts 90 mA aan stroom.
De uitgang van het IC wordt verbonden met een 8 voudige Butterworth filter die slechts frequenties onder de 5Khz doorlaat. Na deze filter volgt de versterker IC4 die rechtstreeks de luidspreker aanstuurd. Het gebruikte kristal moet volgens de datasheet 3.12Mhz zijn. Bij mij werkt echter het goedkopere 4Mhz kristal met goed resultaat. Verder is nog te noemen IC1 een buffer IC die de input data omhoog of omlaag krikt. In figuur 1 is het schema getekend. Figuur 2 geeft de print layout. Om het geheel te laten werken moet de print als fig. 3 worden verbonden met de ATOM. In fig. 4 volgt dan nog de pinaansluitingen.

Vergeet de 4 draadbruggen niet op de print te verbinden alsmede de aansluiting van de speaker. Ook is het verstandig om IC voetjes te gebruiken, vooral voor het spraak IC. Alvorens IC2 de SP 0256 AL2 in de socket te steken als de gehele print vol gesoldeerd is kan men de rest van de print alvast aansluiten. Met de instelpot moet nu de versterker ruis te regelen zijn. Als dit inderdaad het geval is kan men verder gaan. Steek nu IC2 in zijn voet en let hierbij goed op pin 1 en check dat de spanning uit staat.

Naast deze hardware is natuurlijk de software ook onontbeerlijk. Wat moet er nu precies gebeuren. Allereerst moet op pin 2 en 25 de RESET puls worden gegeven. Deze moet volgens de datasheet minstens 100 microseconden lang zijn. Nadat deze RESET puls is aangekomen komt er slechts ruis uit de luidspreker. De schakeling is zo gebouwd dat bij POWER UP eveneens een RESET wordt gegeven. Na het RESET signaal wordt op A1 t/m A6 de data aangeboden en gelatched met ALD. In eerste instantie is ALD 1 evenals RESET. Onmiddellijk hierna wordt ALD gepulsd en begint het spraak IC met de aangegeven foneme. De uitgang SBY wordt laag zodra de voor deze foneme benodigde tijd is verstreken, maar het IC blijft de foneme wel uitspreken. Dit gebeurd net zolang totdat er een nieuwe ALD of RESET puls komt. Met PA2 bekijken we SBY en kunnen we constateren of een nieuwe foneme gezonden kan worden.
Het enige wat nu nog nodig is is de volgorde waarin de fonemen moeten worden verzonden naar het spraak IC. Als we naar de tabel van de fonemenvoorraad van de SP 0256 AL2 kijken dan zien we dat deze op de amerikaanse taal is gebaseerd. Derhalve zal het IC een amerikaans accent hebben. Sommige Nederladse klanken zijn voor het IC moeilijk uit te spreken b.v. de G van sCHeveningen. De verstaanbaarheid van de kunstmatige stem hangt dan ook zeer sterk samen met de foneme keuze. Men moet geen "radio kwaliteit" verwachten, maar de spraak is goed te verstaan. In het bepalen van de foneme is het belangrijk om te luisteren naar de klank en deze te vergelijken met de tabel. Belangrijk is eveneens om een aantal stiltes te plaatsen voor bepaalde fonemen. Zo moet men:

PA1 of PA2 plaatsen voor BB DD GG en JH
PA3 plaatsen voor PP TT KK en CH
NN1 plaatsen voor YR IY EY EH XR AE ER AX AW en UW
NN2 plaatsen voor VV OW OY OR AR en AA

Ook moet men rekening houden met de lengte van bepaalde klanken. b.v. klank 51 ER1 is 110ms en klank 52 ER2 is 210ms. Al met al dus genoeg om te experimenteren.

Na het lezen van dit alles zult u waarschijnlijk aan de slag willen. Listing 1 is bedoeld om de hardware te testen en als alles goed gaat zult u (bij het runnen) de woorden ACORN ATOM SPEECH SYNTHESIZER uit de luidspreker horen. Listing 2 is echter een stuk gecompliceerder en vergt dacht ik enige uitleg.
Allereerst is het programma gebaseerd op de engelse taal. Dit is gedaan omdat de verstaanbaarheid groter is in deze taal. Dit betekent niet dat de ATOM geen nederlands kan spreken maar wel dat er bepaalde aanpassingen noodzakelijk zijn in de wijze waarop text wordt ingevoerd. Men moet nederlands in fonetisch engels invoeren.
Zeer veel mogelijkheden die een dergelijk programma zou moeten hebben zijn echter aanwezig in dit programma.
Bij RUN verschijnt eerst een inleidend plaatje en dan de werk pagina. In het kort worden de hierna beschreven mogelijkheden beschreven. De inleidende stem spreekt u toe en deze text verschijnt tevens op het scherm. Als u hierna iets intijpt gevolgt door spatie of return zal dit onmiddellijk worden uitgesproken.

# = spellen

Als een woord het teken # bevat of een cijfer dan wordt het gehele woord voor u letter voor letter uitgesproken (zo wordt ATOM aah tee ow ehm). Op deze wijze kunnen eigennamen of andere belangrijke woorden worden uitgesproken. Ook zijn aan enkele tekens met name *=-/.,+ woorden toegekend.

@ = fonemencode

Bij computerspraak in de nederlandse taal zijn soms fonemen nodig die normaal gesproken niet worden bereikt met dit programma. Om alle fonemen toegankelijk te maken is deze wijze van invoeren mogelijk gemaakt. De alpha teken beduid dat het daaropvolgende ASCII teken als 6 bit fonemencode geinterpreteerd dient te worden. (Hierbij kunt u het gemakkelijkst de tabel ter hand nemen). Een tweede alpha in het woord beiendigd deze werkwijze.

A-Z = spellen

Als u voor een return of spatie als einde woord slechts een letter of cijfer heeft gegeven dan zal dit teken eveneens in het geheel worden uitgesproken. b.v. W geeft DUBEL;JO-UW.

REPT + TOETS = herhalen

Als u de REPT toets tesamen met een andere toets indrukt dan wordt het laatste wat de speech synthesizer heeft gezegt herhaald.

CTRL + TOETS = voorgeprogrammeerde text.

Als u tesamen met de CONTROL toets een andere ingedrukt houdt dan zal u een van de 8 voorgeprogrammeerde texten horen. Welke is afhankelijk van de lage 3 bits van het ASCII code van de ingedrukte toets. Om te voorkomen dat bepaalde control opdrachten direct worden uitgevoerd is het verstandig om samen met control alleen de toetsen 0 t/m 9 in te drukken. U kunt hiermee de gehele voorgeprogrammeerde woordenvoorraad bereiken.

Het programma is met opzet zodanig in elkaar gezet dat men eenvoudig wijzigingen kan aanbrengen. Om het op 2 kantjes A4 te krijgen is echter gebruik gemaakt van een verlengde inputbuffer zodat als u deze niet heeft u soms een tussenregel zult moeten inlassen.
De individuele letters worden uitgesproken zoals aangegeven in de tabel die begint op regel 740 en eindigt op regel 760. #02 is dus de fonemencode voor @ #3B voor A enz. Ook begint op regel 760 de tabel voor lettercombinatie aanpassingen. Zo zien we op regel 890 dat PH moet worden uitgesproken als #A8. Dit getal achter de letter combinaties is altijd #80 groter dan de fonemencode om onderscheid te maken met normale ASCII tekens. De fonemencode is dus #28 hetgeen FF is. Deze tabel eindigd op regel 920. Op regel 930 start de tabel die nodig is voor het spellen. We zien hier hoe alle klanken moeten worden gevormd. Deze tabel heeft als einde regel 1140. Tevens begint op deze regel de laatste tabel voor de voorgeprogrammeerde texten. Deze tabel eindigd op regel 1170.

| | | | | | | |
|---|---|---|---|---|---|---|
| 00 | 00 | PA1 | @ | | PAUSE | 10 |
| 01 | 01 | PA2 | A | | PAUSE | 30 |
| 02 | 02 | PA3 | B | | PAUSE | 50 |
| 03 | 03 | PA4 | C | | PAUSE | 100 |
| 04 | 04 | PA5 | D | | PAUSE | 200 |
| 05 | 05 | OY | E | | boY | 290 |
| 06 | 06 | AY | F | | skY | 170 |
| 07 | 07 | EH | G | E | End | 50 |
| 08 | 08 | KK3 | H | | Comb | 80 |
| 09 | 09 | PP | I | P | Pow | 150 |
| 0A | 10 | JH | J | | dodGe | 100 |
| 0B | 11 | NN1 | K | N | thiN | 170 |
| 0C | 12 | IH | L | | sIt | 50 |
| 0D | 13 | TT2 | M | T | To | 100 |
| 0E | 14 | RR1 | N | R | Rural | 130 |
| 0F | 15 | AX | O | | sUcceed | 50 |
| 10 | 16 | MM | P | M | Milk | 180 |
| 11 | 17 | TT1 | Q | | parT | 80 |
| 12 | 18 | DH1 | R | | THey | 140 |
| 13 | 19 | IY | S | | sEE | 170 |
| 14 | 20 | EY | T | I | bEIge | 200 |
| 15 | 21 | DD1 | U | | coulD | 50 |
| 16 | 22 | UW1 | V | | tO | 60 |
| 17 | 23 | AO | W | O | AUght | 70 |
| 18 | 24 | AA | X | | hOt | 60 |
| 19 | 25 | YY2 | Y | J, Y | Yes | 130 |
| 1A | 26 | AE | Z | | hAt | 80 |
| 1B | 27 | HH1 | [ | H | He | 90 |
| 1C | 28 | BB1 | \ | B | Busy | 40 |
| 1D | 29 | TH | ] | | THin | 130 |
| 1E | 30 | UH | ↑ | U | bOOk | 70 |
| 1F | 31 | UW2 | | | fOOd | 170 |

| | | | | | | |
|---|---|---|---|---|---|---|
| 20 | 32 | AW | ` | | OUt | 250 |
| 21 | 33 | DD2 | a | D | Do | 250 |
| 22 | 34 | GG3 | b | | wiG | 120 |
| 23 | 35 | VV | c | V | Vest | 130 |
| 24 | 36 | SH | d | G | Guest | 80 |
| 25 | 37 | SH | | | SHip | 120 |
| 26 | 38 | ZH | | | aZure | 130 |
| 27 | 39 | RR2 | g | | bRain | 80 |
| 28 | 40 | FF | h | F | Food | 110 |
| 29 | 41 | KK2 | i | | sKy | 140 |
| 2A | 42 | KK1 | j | K, C, Q | Can't | 120 |
| 2B | 43 | ZZ | k | Z | Zoo | 150 |
| 2C | 44 | NG | l | | aNchor | 200 |
| 2D | 45 | LL | m | L | Lake | 80 |
| 2E | 46 | WW | n | W | Wool | 140 |
| 2F | 47 | XR | o | | repAIR | 250 |
| 30 | 48 | WH | p | | WHig | 150 |
| 31 | 39 | YY1 | q | | Yes | 90 |
| 32 | 50 | CH | r | | CHurch | 150 |
| 33 | 51 | ER1 | s | | fIR | 110 |
| 34 | 52 | ER2 | t | | fIR | 210 |
| 35 | 53 | OW | u | | bEAU | 170 |
| 36 | 54 | DH2 | v | | THey | 180 |
| 37 | 55 | SS | w | X, S | veSt | 60 |
| 38 | 56 | NN2 | x | | No | 140 |
| 39 | 57 | HH2 | y | | Hoe | 130 |
| 3A | 58 | OR | z | | stORe | 240 |
| 3B | 59 | AR | { | A | alARm | 200 |
| 3C | 60 | YR | \| | | clEAR | 250 |
| 3D | 61 | GG2 | } | | Got | 80 |
| 3E | 62 | EL | ~ | | saddLE | 140 |
| 3F | 63 | BB2 | | | Busy | 60 |

stuklijst

halfgeleiders

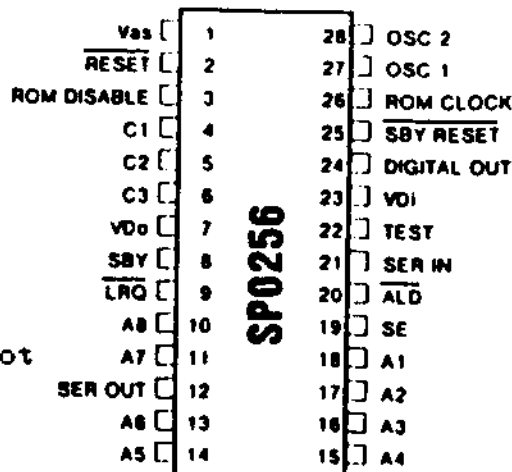
1 SP 0256-AL2
1 74 LS 541
1 LM 386
1 LM 324
1 1N4148

condensatoren

2 22 pF
8 10 nF
8 100 nF
1 10 uF elco
1 100 uF elco

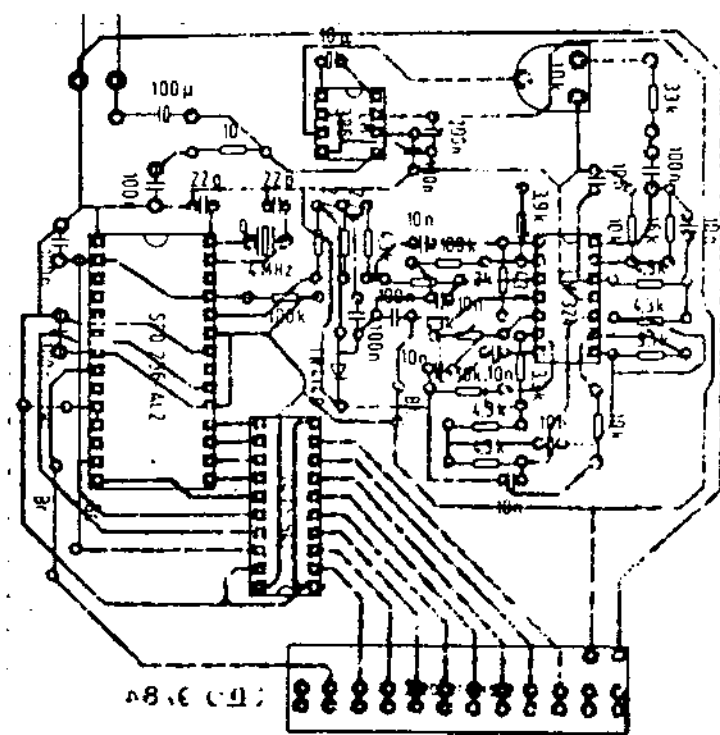
weerstanden

1 10 ohm
2 2.7 kohm
1 3.3 kohm
1 3.9 kohm
8 4.3 kohm
1 9.1 kohm
3 10 kohm
1 16 kohm
1 33 kohm
2 100 kohm
1 10 kohm instelpot



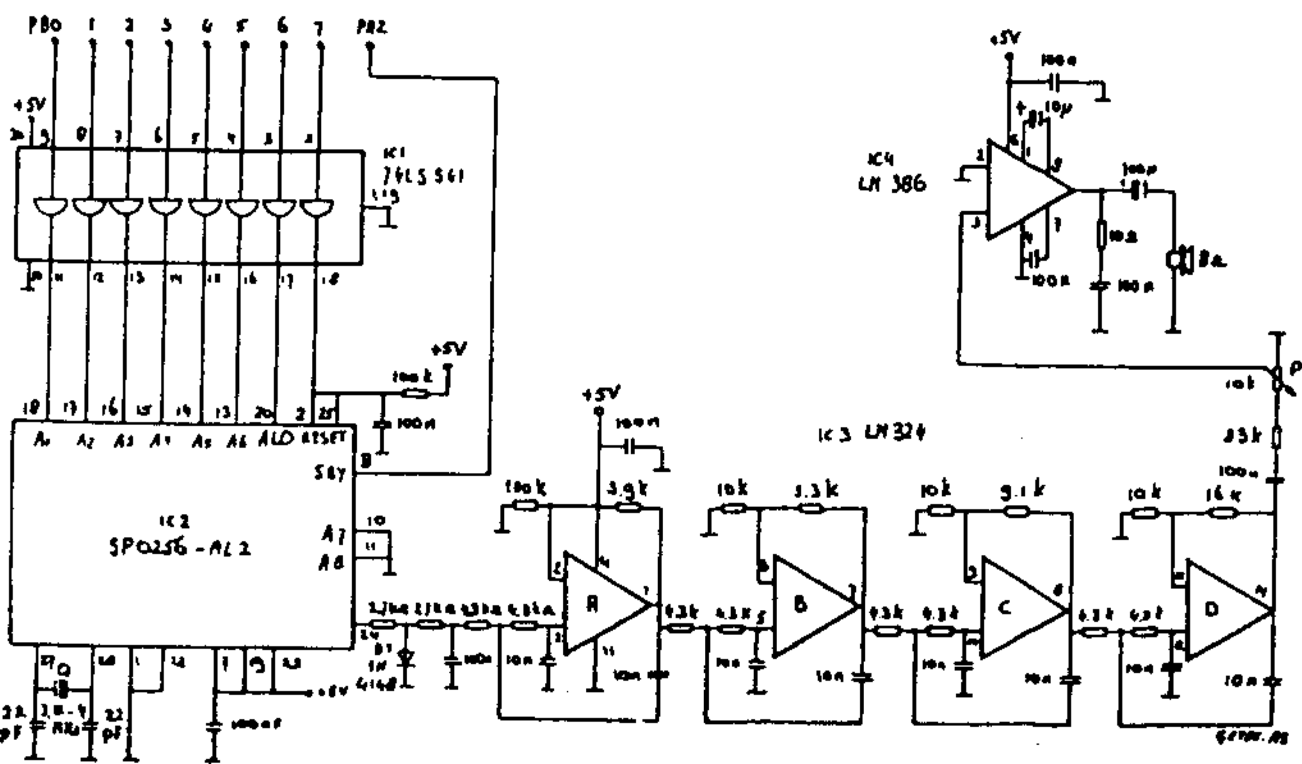

overige

1 kristal 3.12 Mhz - 4.00 Mhz
1 connector met de ATOM B poort



## PINFUNCTIES

| Pin Nummer | Naam | Functie |
|---|---|---|
| 1 | $V_{ss}$ | Aarding |
| 2 | $\overline{RESET}$ | Een logische 0 stelt de SP terug op nul. Moet naar een logische 1 omgeschakeld worden voor normale werking |
| 3 | ROM DISABLE | Voor gebruik met een externe seriële spraak-ROM. Een logische 1 schakelt de externe ROM uit. |
| 4, 5, 6. | C1, C2, C3, | Uitgangs-controlelijnen gebruikt door een externe seriële spraak-ROM |
| 7 | $V_{DD}$ | Hoofdvoeding |
| 8 | SBY | STANDBY. De uitvoer van een logische 1 geeft aan dat de SP buiten werking is (d.i. niet aan het spreken) en $V_{DD}$ kan extern uitgeschakeld worden om het verbruik te beperken. Wanneer de SP weer bekrachtigd wordt door een adres dat geladen wordt, schakelt SBY om naar een logische 0. |
| 9 | $\overline{LRQ}$ | $\overline{\text{LOAD REQUEST}}$. $\overline{LRQ}$ is de uitvoer van een logische 1, telkens wanneer de invoerbuffer vol is. Wanneer $\overline{LRQ}$ naar een logische 0 omschakelt, wordt de invoerpoort geladen door de 8 adresbits op A1-A8 te zetten en een puls te geven op de $\overline{ALD}$ ingang. |
| 10, 11, 13, 14<br>15, 16, 17, 18 | A8, A7, A6, A5<br>A4, A3, A2, A1 | 8-bits adres dat een van de 256 punten voor spraakinvoer bepaalt |
| 12 | SER OUT | SERIAL ADRESS OUT. Deze uitgang draagt een 16-bits adres serieel over naar een externe spraak ROM. |
| 19 | SE | STROBE ENABLE. Normaal gezien op logisch 1 gehouden. Wanneer verbonden met de aarding, wordt $\overline{ALD}$ uitgeschakeld en vergrendeld de SP automatisch het adres op de invoerbus gedurende ongeveer 1µs, na detectie van een logische 1 op een adreslijn. |
| 20 | $\overline{ALD}$ | $\overline{\text{ADDRESS LOAD}}$. Een negatieve puls op deze ingang laadt de 8 adresbits in de ingangspoort. Door de stijghelling van deze puls schakelt $\overline{LRQ}$ om naar een logische 1. |
| 21 | SER IN | SERIAL IN. Een seriële invoer van een 8-bits gegeven vanuit een externe spraak-ROM. |
| 22 | TEST | Een logische 1 brengt de SP in de testmodus. Deze pin moet normaal gezien geaard zijn. |
| 23 | V | Noodvoeding voor interfacelogica en controller. |
| 24 | DIGITAL OUT | Impulsiebreedte-gemoduleerde digitale spraakuitgang, die bij filtering door een 5KHz laagdoorlaatfilter en versterking een luidspreker aan drijft. |
| 25 | $\overline{SBY\ RESET}$ | $\overline{\text{STANDBY RESET}}$. Een logische 0 stelt de interfacelogica terug op nul. Moet normaal gezien een logische 1 zijn. |
| 26 | ROM CLOCK | 1,56 MHz klok voor een externe seriële spraak-ROM |
| 27 | OSC 1 | XTAL IN. Ingangsaansluiting voor een 3,12 MHz kristal |
| 28 | OSC 2 | XTAL OUT. Uitgangsaansluiting voor een 3,12 MHz kristal |



```
 10REM ***************************
 20REM **                       **
 30REM **    TYPE EN TALK       **
 40REM **                       **
 50REM **     B  KASTEEL        **
 60REM **     070 202314        **
 70REM **                       **
 80REM ***************************
 90DIM LL(80);F.I=0TO80;LL(I)=-1;N.;P.$21;F.I=0TO1;?#E8=0;P=#2C00;[
100JSR#F7D1;];!P=#0C0A;P=P+2
110$P="             TYPE EN TALK";P=P+L.P;!P=#0A0A0D;P=P+3
120$P=" # OF CIJFER IN WOORD = SPELLEN";P=P+L.P;!P=#0D0A;P=P+2
130$P=" LETTERS TUSSEN @ = ASCII CODE ";P=P+L.P;!P=#0D0A;P=P+2
140$P=" ALLEEN A - Z OF 0 - 9 SPELLEN ";P=P+L.P;!P=#0D0A;P=P+2
150$P=" CTRL+TOETS IS VOORGEPROG. TEXT";P=P+L.P;!P=#0D0A;P=P+2
160!P=#0D0A0A0A;P=P+4;!P=#0A0A0A0A;P=P+4
170$P="             B  KASTEEL";P=P+L.P;!P=#0D0A;P=P+2
180$P="             070 202314";P=P+L.P;[;NOP
190LDX@#DF;:LL69;LDA#8000,X;ORA@#80;STA#8000,X;DEX;BNELL69
200LDA@#A0;STA#8000;STA#8180
210LDX@#7F;:LL70LDA#8180,X;ORA@#80;STA#8180,X;DEX;BNELL70;SEI
220CLD;LDX@#FF;TXS;LDA@2;STA8;LDA@0;STA#B803;LDA@#FF;STA#B802
230LDA@192;STA#B800;LDA@64;STA#B800;LDA@192;STA#B800
240LDA@#15;STA#80;STA#81;LDA@#04;JSR LL53;JSR LL50;JMP LL4
250:LL75PLP;PLA;JSRLL50;LDY@#01
260:LL78LDA#8E,Y;CMP@#2D;BEQLL79;JSRLL9;INY;BNELL78
270:LL79LDA@13;JSRLL9;JMPLL1;:LL1JSRLL55;:LL2;:LL3
280JSR LL58;STA#08;PHA;PHP;LDA#B002;AND@#40;BEQLL75
290LDA#B001;AND@#40;BEQLL4;PLP;PLA
300PHA;PHP;TXA;PHA;LDA#85;CMP@#01;BNELL68;JSRLL66
310:LL68PLA;TAX;PLP;PLA;JSRLL9;JMPLL2;:LL4LDA@D&#FF;STA#8A
320LDA@D/256&#FF;STA#8B;LDY@0;LDA#08;AND@#07;TAX;BEQ LL7
330:LL5LDA(#8A),Y;BEQ LL1;INY;BNE LL6;INC#8B;:LL6CMP@#0D;BNELL5;DEX;BNELL5
340:LL7LDA(#8A),Y;BEQLL1;JSRLL9;INY;BNELL8;INC#8B;:LL8CMP@#0D;BNELL7;BEQLL1
350:LL9STA#89;TXA;PHA;TYA;PHA;LDA#89;AND@#7F;LDX#85;CPX@#40;BCCLL10;JMPLL27
360:LL10STA#8E,X;CMP@#21;BCCLL15;PHA;PHP;TXA;PHA
370LDA#8E,X;CLC;ADC@#20;AND@#7F;EOR@#60;STA#811F,X;INX
380:LL71LDA@#20;STA#811F,X;INX;CPX@#40;BNELL71;PLA;TAX;PLP;PLA
390INC#85;CMP@#24;BEQ LL14;CMP@#40;BEQLL14;CMP@#23;BEQLL11;CMP@#3A;BCSLL13
400CMP@#30;BCC LL13;:LL11BIT#84;BMILL13;:LL12LDA@#80;STA#82
410:LL13PLA;TAY;PLA;TAX;LDA#89;RTS;:LL14LDA@#80;STA#84;BNELL13;:LL15
420LDA@#2D;STA#8E,X;CPX@#02;BEQLL16;BIT#82;BPL LL27;:LL16STX#86
430:LL17LDA#8E,X;STA#9E,X;DEX;:LL18BNELL17;JSRLL50;LDX@#01
440:LL19LDA#9E,X;ORA@#80;LDY@#00;:LL20CMP C,Y;BEQLL21;INY;BNELL20;BEQLL22
450:LL21INY;LDA C,Y;BMILL22;JSR LL9;:LL57JMPLL21
460:LL22LDA@#20;JSRLL9;INX;CPX#86;BCCLL19;:LL23BCSLL13;:LL24LDA#83
470:LL25EOR@#80;STA#83;JMPLL42
480:LL26INX;LDA#8E,X;AND@#03;ASLA;ASLA;ASLA;ASLA;STA#87
490INX;LDA#8E,X;AND@#0F;ORA#80;STA#80;JMPLL42;:LL27LDX@#00
500:LL28LDA#8E,X;CMP@#40;BEQLL24;CMP@#24;BEQLL26;STX#87
510LDA#80;CLC;ADC B+7;STA#81;DEX;:LL29JSRLL44;BMILL29;:LL30JSRLL44;BPL LL30
520:LL31JSRLL44;BMILL31;:LL32LDX#87;LDA#8E,X;BIT#83;BPLLL34;JSRLL51
530:LL33JMPLL42;:LL34LDY@#00;STY#88;:LL35LDA F,Y;BMILL43
540:LL36CMP#8E,X;BNELL37;INX;INY;BNELL35;:LL37LDX#87
550:LL38INY;LDA F,Y;BPLLL38;:LL39INY;LDA F,Y;BMILL39
560:LL40STY#88;CMP@#00;BNELL36
570:LL41LDA#8E,X;CMP@#41;BCCLL42;AND@#3F;TAY;LDA E,Y;JSRLL51
580:LL42INX;CPX#85;BCCLL28;JSRLL50;JMPLL13
590:LL43DEX;LDA F,Y;BPLLL42;JSRLL51;INY;BNELL43+1
600:LL44INX;LDA#8E,X;CMP@#2D;BEQLL49;LDY@#06
610:LL45CMP B,Y;BEQLL48;DEY;BPLLL45;LDY@#04
620:LL46CMP B+#E,Y;BEQ LL47;DEY;BPL LL46;:LL47RTS
```

```
630:LL48LDA B+7,Y;CLC;ADC#80;STA#81;:LL49PLA;PLA;JMP LL32
640:LL50LDA@#03;JSRLL51;LDA@#00;STA#82;STA#83;STA#84;LDA@#2D
650STA#BE;LDA@#01;STA#85;RTS;:LL51AND@#3F
660PHA;:LL52LDA#B801;AND@#04;BEQ LL52;PLA
670:LL53PHA;CLC;ADC@192;STA#B800;PLA;PHA;CLC;ADC@128;STA#B800
680PLA;CLC;ADC@192;STA#B800;RTS;:LL55LDA@#8A;STA#B003;LDA@#07;STA#B002;RTS
690:LL58PHP;CLD;STX#E4;STY#E5;:LL59BIT#B002;BVC LL60;JSR#FE71;BCC LL59
700:LL60JSR#FB8A;:LL61
710JSR#FE71;BCSLL61;JSR#FE71;BCSLL61;JMP#FEB1
720:LL62PLA;STA#08;PLP;LDX#E4;LDY#E5;PLP;PLA;PLA;JMP LL4
730:LL66LDX@#40
740:LL67LDA@#20;STA#811F,X;DEX;BNELL67;RTS;];B=P;$P=" .:,:?!";P=P+L.P
750!P=#01010002;P=P+4;!P=#040301;P=P+3;$P="AEIOU";P=P+L.P;E=P
760!P=#2A1C3B02;P=P+4;!P=#24280721;P=P+4;!P=#2A19131B;P=P+4
770!P=#170B102D;P=P+4;!P=#320E2A09;P=P+4;!P=#2E231E0D;P=P+4
780!P=#022B1937;P=P+4;!P=#04040404;P=P+4;F=P;$P="EI";P=P+L.P;!P=#94;P=P+1
790$P="AE";P=P+L.P;?P=#9A;P=P+1;$P="OU";P=P+L.P;!P=#A0;P=P+1
800$P="AA";P=P+L.P;!P=#94;P=P+1;$P="EE";P=P+L.P;!P=#93;P=P+1
810$P="OY";P=P+L.P;!P=#B5;P=P+1;$P="AY";P=P+L.P;!P=#86;P=P+1
820$P="EH";P=P+L.P;!P=#87;P=P+1;$P="TH";P=P+L.P;!P=#92;P=P+1
830$P="AU";P=P+L.P;!P=#A0;P=P+1;$P="OO";P=P+L.P;!P=#9F;P=P+1
840$P="SH";P=P+L.P;!P=#A5;P=P+1;$P="AIR";P=P+L.P;!P=#AF;P=P+1
850$P="WH";P=P+L.P;!P=#B0;P=P+1;$P="WO";P=P+L.P;!P=#AE9E97;P=P+3
860$P="IR";P=P+L.P;!P=#B3;P=P+1;$P="EAU";P=P+L.P;!P=#B5;P=P+1
870$P="OR";P=P+L.P;!P=#BA;P=P+1;$P="EAR";P=P+L.P;!P=#BC;P=P+1
880$P="IE";P=P+L.P;?P=#93;P=P+1;$P="SCH";P=P+L.P;!P=#AAB7;P=P+2
890$P="EL";P=P+L.P;?P=#BE;P=P+1;$P="LE";P=P+L.P;?P=#BE;P=P+1
900$P="CH";P=P+L.P;?P=#B2;P=P+1;$P="NG";P=P+L.P;!P=#AC;P=P+1
910$P="PH";P=P+L.P;?P=#A8;P=P+1;$P="ER";P=P+L.P;!P=#B3;P=P+1
920$P="X";P=P+L.P;!P=#B7AA;P=P+2;$P="AR";P=P+L.P;!P=#BB;P=P+1
930$P="LL";P=P+L.P;?P=#AD;P=P+1;$P="MM";P=P+L.P;?P=#90;P=P+1
940$P="[";P=P+L.P;!P=#8484;P=P+2;!P=#0000;P=P+2;C=P
950$P=" ZEEROW";?P=#B0;P=P+L.P;$P=" WAN";?P=#B1;P=P+L.P
960$P=" TWO";?P=#B2;P=P+L.P;$P=" T-HREE";?P=#B3;P=P+L.P
970$P=" FO-UR";?P=#B4;P=P+L.P;$P=" FAIV";?P=#B5;P=P+L.P
980$P=" SIX";?P=#B6;P=P+L.P;$P=" SE-EVE-EN";?P=#B7;P=P+L.P
990$P=" AAT";?P=#B8;P=P+L.P;$P=" NAIN";?P=#B9;P=P+L.P
1000$P=" EAA";?P=#C1;P=P+L.P;$P=" BEE";?P=#C2;P=P+L.P
1010$P=" SEE";?P=#C3;P=P+L.P;$P=" DEE";?P=#C4;P=P+L.P
1020$P=" EE";?P=#C5;P=P+L.P;$P=" E-EF";?P=#C6;P=P+L.P
1030$P=" DJEE";?P=#C7;P=P+L.P;$P=" AATCH";?P=#C8;P=P+L.P
1040$P=" AI";?P=#C9;P=P+L.P;$P=" DJAA";?P=#CA;P=P+L.P
1050$P=" KEAA";?P=#CB;P=P+L.P;$P=" E-EL";?P=#CC;P=P+L.P
1060$P=" E-EM";?P=#CD;P=P+L.P;$P=" E-EN";?P=#CE;P=P+L.P
1070$P=" OW";?P=#CF;P=P+L.P;$P=" PEE";?P=#D0;P=P+L.P
1080$P=" KJUW";?P=#D1;P=P+L.P;$P=" AR-";?P=#D2;P=P+L.P
1090$P=" ES";?P=#D3;P=P+L.P;$P=" TEE";?P=#D4;P=P+L.P
1100$P=" JO-UW";?P=#D5;P=P+L.P;$P=" VEE";?P=#D6;P=P+L.P
1110$P=" DUBEL;JO-UW";?P=#D7;P=P+L.P;$P=" IK@w@";?P=#D8;P=P+L.P
1120$P=" WAY";?P=#D9;P=P+L.P;$P=" ZET";?P=#DA;P=P+L.P
1130$P=" TAIMS";?P=#AA;P=P+L.P;$P=" AND";?P=#AB;P=P+L.P
1140$P=" KOMA";?P=#AC;P=P+L.P;$P=" LAIN";?P=#AD;P=P+L.P
1150$P=" POINT";?P=#AE;P=P+L.P;$P=" LAIN";?P=#AF;P=P+L.P
1160$P=' IQWALS";?P=#BD;P=P+L.P;?P=#FF;P=P+1;D=P;$P="OWKEY]";P=P+L.P+1
1170$P="SPEECH]";P=P+L.P+1;$P="AACORN;ATOM;SPEECH;SINTHSESAISER]";P=P+L.P+1
1180$P="TEST]";P=P+L.P+1;$P="ERROR]";P=P+L.P+1;$P="AACORN;ATOM]";P=P+L.P+1
1190$P="REESET]";P=P+L.P+1;$P="HALLOW]";P=P+L.P+1;?P=#00;P=P+1
1200N.;P.$6;IF?#E8=35;P."WARNING OUT OF RANGE DETECTED"'
1210P.$12''''"              type-en-talk"'''
1220@=0;P."        VAN #2C00 TOT #"&P''';@=8
1230P."            PRESS ANY KEY"'''$#3E;LINK#FFE3;LINK#2C00;END
```

Hoera!, er komt schot in. Wij (zijnde de hardware commissie) hebben op zondag 16 December een vergadering gewijd aan de vraag "80 koloms moet, maar hoe??". Ook is een voorstel tot uitbreiden (=indelen) voor IN/OUT besproken. Hierbij een kort verslag van de uitkomsten van dit overleg.
Aan dit overleg hebben meegedaan:

| | |
|---|---|
| Rudy van Drunen | regio noord. |
| Peter Ehrlich | regio brabant. |
| Wim Ernst | regio limburg. |
| Leon Heesakkers | regio limburg. |
| Bram Poot | regio twente. |

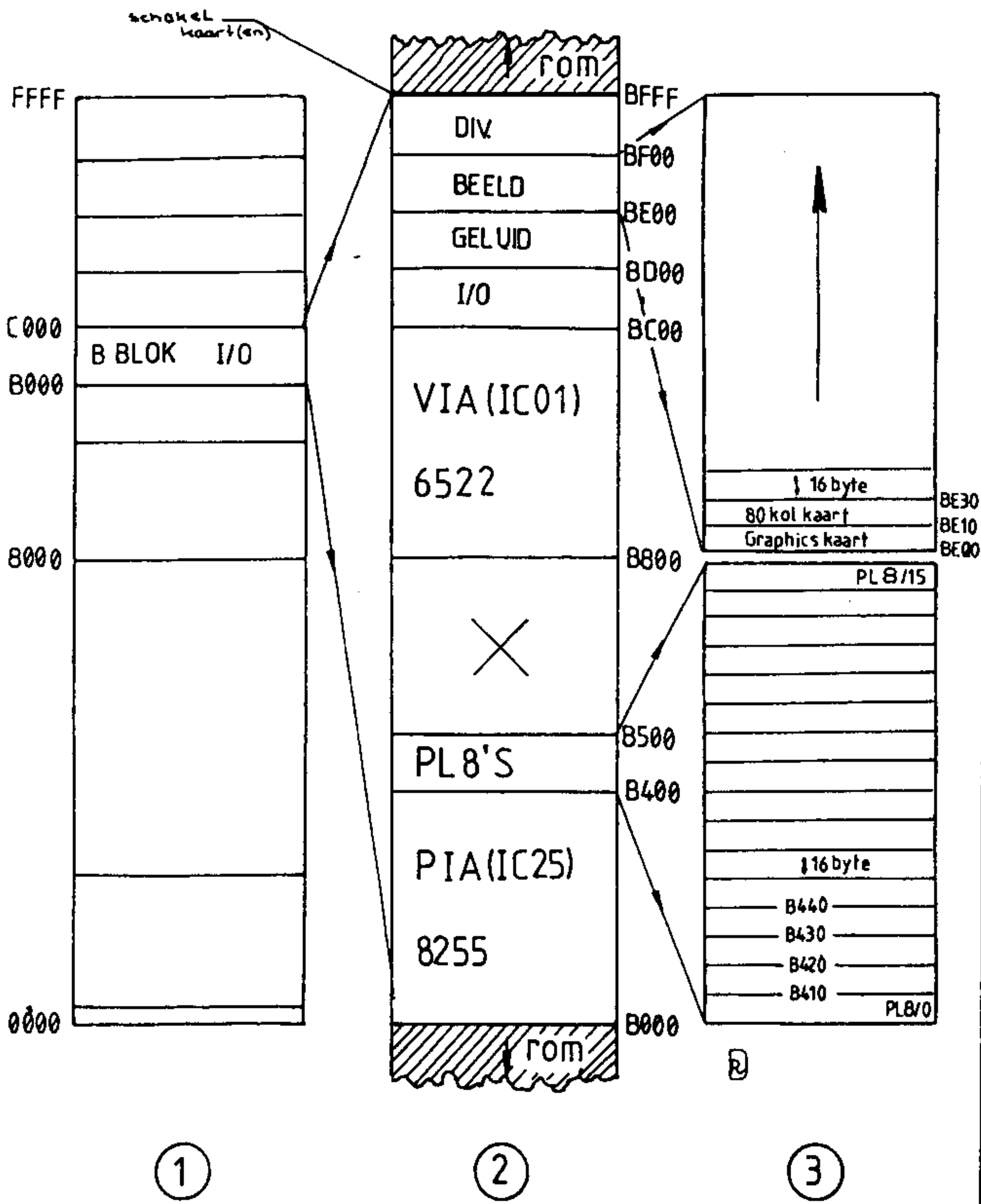
## H E T U I T B R E I D E N = I N D E L E N.

In onze club bestaat een grote drang tot het uitbreiden van onze computer. Dit symptoom schijnt acornisten (ookwel Acornuiten) eigen te zijn.
Het uitbreiden van het geheugen geeft over het algemeen geen grote INDELINGS problemen.
Het I/O uitbreiden heeft wat meer voeten in de aarde. Er is bij ons in de computer een heel 4K blok (#B000-#BFFF) gereserveerd voor input-output. Wanneer we nu de I/O gaan uitbreiden, zullen we de verschillende I/O ic's (bijv. een VIA, PIA, ACIA, enz.) in dit blok gaan adresseren. De vraag is nu: Waar, binnen dit 4K blok, adresseren we nu onze I/O ic's

Wanneer hier door de club geen beleid wordt gevoerd, zal het geheel uitlopen in een chaos. Wij (hardware commissie) hebben hieroven een voorstel gedaan. Dit voorstel wil ik hierbij toelichten.

EEN OPMERKING VOORAF: HET IS NIET VERPLICHT DIT VOORSTEL OVER TE NEMEN, WAT U DOET IS UW ZAAK. DE CLUB WIL HIERMEE VOORKOMEN DAT WE OP EEN GEGEVEN MOMENT VASTLOPEN.

In fig. 1 is de memory-map geschetst, het gebied wat we gaan indelen, is in fig. 2 uitvergroot.
We zien hier dat we het gebied van #B400-#B4FF reserveren voor PL8 connectoren. Deze connectoren zijn primair bestemd voor EIGEN ONTWIKKELINGEN, bijv. een extra VIA. Elke connector heeft 16 bytes toegewezen gekregen. Er zal een gebufferd decoderings kaartje komen ('PL8 kaart') waarop deze 16 connectoren uitgevoerd zijn.
Het gebied van #B500 tot #B7FF. Hiervoor hebben we nog niets vastgesteld. Eventueel te gebruiken voor later, als we echt vol zitten.
Het gebied van #BC00 tot #BCFF is gereserveerd voor CLUBKAARTEN, die iets met I/O doen, bijv. een RS232c kaart, een FLOPPY kaart ,enz.
Evenzo reserveren we #BD00-BDFF voor CLUBKAARTEN, die met GELUID te maken hebben, te denken valt hier aan een soundboard, of speechkaart
Het gebied van #BE00 tot #BEFF reserveren we voor BEELD kaarten. bijv. de 80 koloms kaart of graphics kaart. (zie elders in dit nummer)
Het gebied van #BF00 tot #BFFF tenslotte, zal gebruikt worden vor CLUBKAARTEN, die niet in bovenstaande categorieen vallen. bijv de schakelkaart (schakelbyte)




-facc- —de voorgestelde indeling-

In fig. 3 zien we de opdeling van de gebiedjes. Hier geldt steeds: dekoderen tot gebieden van 16 bytes. Wanneer een systeem bijv. alleen een 80 kol. kaart bevat, dan is de display controller geadresseerd op bijv. #BE10-#BE1F, de rest blijft dan leeg. De CLUBKAARTEN hebben dus een vaste plaats. Dit geeft ook geen problemen met software.

**NAWOORD:**

Deze indeling zal door de club op de CLUBKAARTEN toegepast worden. Het is ABSOLUUT niet de bedoeling dat u dit voorstel slaafs volgt. Wanneer u er niet mee eens bent kunt u twee dingen doen:

1) Uw eigen zin doorvoeren, en dan de eventueel aan te schaffen clubkaarten ombouwen, geen probleem voor ons.
2) Uw mening kenbaar maken aan ons:

HARDWARE COMMISSIE
POSTBUS 1050
9700 BA GRONINGEN.

Dan kunnen we er nog eens over praten, of we het wellicht anders moeten doen.

# D E 8 0 K O L O M S K A A R T

## 1.0 WAT DOEN WE?

Er is gesignaleerd, dat er binnen de club plaats is voor twee versies van de 80 kolomskaart, nl. een versie die alleen 80 karakters weer kan geven en een versie die bovendien de mogelijkheid van graphics met een hogere resolutie heeft. Deze 'graphics VDU kaart' zal, voorzover dit technisch mogelijk is ook in verschillende uitvoeringen op te bouwen zijn.

## 2.1 ONTWIKKELING

De kaart die alleen de mogelijkheid van 80 (40) kolommen bezit is ontwikkeld door Chris Kwakernaak. Momenteel wordt er nog wat aan het ontwerp bijgeschaafd, om de print daarna te ontwerpen.
Bij de grafische kaart staan de zaken er wat anders voor: Omdat het hier over een vrij complexe schakeling gaat vraagt dit ook een wat langere ontwikkelingstijd. De kaart zal gebaseerd zijn op het GDP64 ontwerp van GRAF (zie A.N. 7-'84), pag. 75-77

## 2.2 TERMIJN

De grote vraag is nu dus "WANNEER IS HET KLAAR??". Nu is dit, omdat wij geen professionals zijn, natuurlijk zeer moeilijk te zeggen. Wel is zeker dat de tekstkaart eerder klaar zal zijn dan de kaart met grafische mogelijkheden. Een ZEER VOORZICHTIGE SCHATTING is voor de tekstkaart midden '85.

## 2.3 PRIJS

Dit is een zeer gevoelig punt, mede omdat we met zeer SCHOMMELENDE IC PRIJZEN te maken hebben. Ook hier een ZEER VOORZICHTIGE SCHATTING: De eenvoudigste uitvoering tekstkaart +/- F.200.-, De grafische kaart, afhankelijk van de uitvoering van +/- F.275 voor de eenvoudigste, tot plm. F.450.- voor de meest geavanceerde.

2.4 OPMERKING

DE HIERGEGEVEN PRIJZEN, EN ONTWIKKELINGS TERMIJNEN ZIJN ONDER ZEER GROOT VOORBEHOUD!!, ze zijn bedoeld als RICHTWAARDEN.

3.0 MOGELIJKHEDEN

Er komen dus twee kaarten een tekstkaart, en een kaart die ook grafisch een woordje meespreekt.
Van de eerste is reeds nauwkeurig bekend wat de mogelijkheden zijn. Van de grafische kaart kan ik alleen nog maar een GLOBALE indruk geven.

3.1.0 TEKSTKAART

Eerst een beknopte beschrijving van de 'tekstkaart'.
De kaart zal op EUROKAART worden uitgevoerd worden, waarbij de 10*16 cm niet geheel voor de schakeling gebruikt zal worden. De overgebleven ruimte zal worden opgevuld door een veld met soldeer eilandjes.

3.1.1 HARDWARE

De gebruikte display controller is van het type EF 9345, van THOMSON EFCIS. Deze controller heeft een ingebouwde karaktergenerator. Een externe karakter (E)PROM, of RAM is op de print niet voorzien, maar is wel mogelijk. De VDU RAM bestaat in een minimum uitvoering uit een 6116 (2K). De RAM kam worden uitgebreidt tot 6, of 12 Kbyte. Dit vergroot de mogelijkheden, waarover straks meer. Op de print is ruimte voor 3 6116's (6K), wie meer wil moet dus stapelen. De VDU RAM is niet in de memory-map geplaatst, het is dus niet direct uileesbaar. De scrolling gaat hardwarematig. Er is in een schakeling die bij omschakeling naar graphics, het videosignaal omschakelt naar het originele (computer) signaal voorzien.

3.1.2 DE MOGELIJKHEDEN

25 regels met, naar keuze 40 of 80 karakters
-Met 2K VDU geheugen:
Bij 40 karakters: per regel instelbaar oa. dubbele breedte, cq. hoogte, onderstrepen, omgekeerd, half-intensity.
Bij 80 karakters: geen extra mogelijkheden.
-Met 6K VDU geheugen
Bij 40 karakters, zie boven, maar dan per letter individueel instelbaar
Bij 80 karakters: knipperend, onderstreept, verschillende helderheden.

De ingeboude karakter ROM bevat ook 128 blok graphics symbolen. Verder zijn in de ASCII set van 128 karakters, enige speciale karakters opgenomen.

3.1.3 DE SOFTWARE

Deze is ontwikkeld door Chris Kwakernaak. Hieraan moet ook nog wat

### 3.1.3 DE SOFTWARE

Deze is ontwikkeld door Chris Kwakernaak. Hieraan moet ook nog wat bijgeschaaft worden. De lengte zal de 2K niet overschrijden. De software kan overal 'neergezet' worden.

## 3.2 DE GRAPHICSKAART

Omdat de grafische kaart voor een groot gedeelte nog ontwikkeld moet worden is hier nog niet zoveel over bekend.

Het zal een print worden (eurokaat) waarop 3 verschillende uitvoeringen van de graphicskaart gebouwd zullen kunnen worden. Het verschil in de 3 uivoeringen zit 'm voornamelijk in de maximale grafische resolutie (oplossend vermogen) en de toegepaste display-pocessor

bijv.
1. Kaart met EF9366, max.512*256 punten, 16K dyn. RAM
2. Kaart met EF9365, max.512*512 punten, 64K dyn. RAM, 2 schermen
3. Kaart met EF9367, max.1024*512 punten, 64K dyn. RAM

De kaart zal worden voorzien van HARDWARE SCROLLING, en een mogelijkheid om het beeldschermgeheugen uit te lezen.
Voor de 16K versie zal er op de print een omvormer voor de -5, en de +12V voor de 16K rams gemaakt moeten worden.
Ook in deze controllers is een ROM voor de karakteropbouw ingebouwd.

## 4. MONITOR?

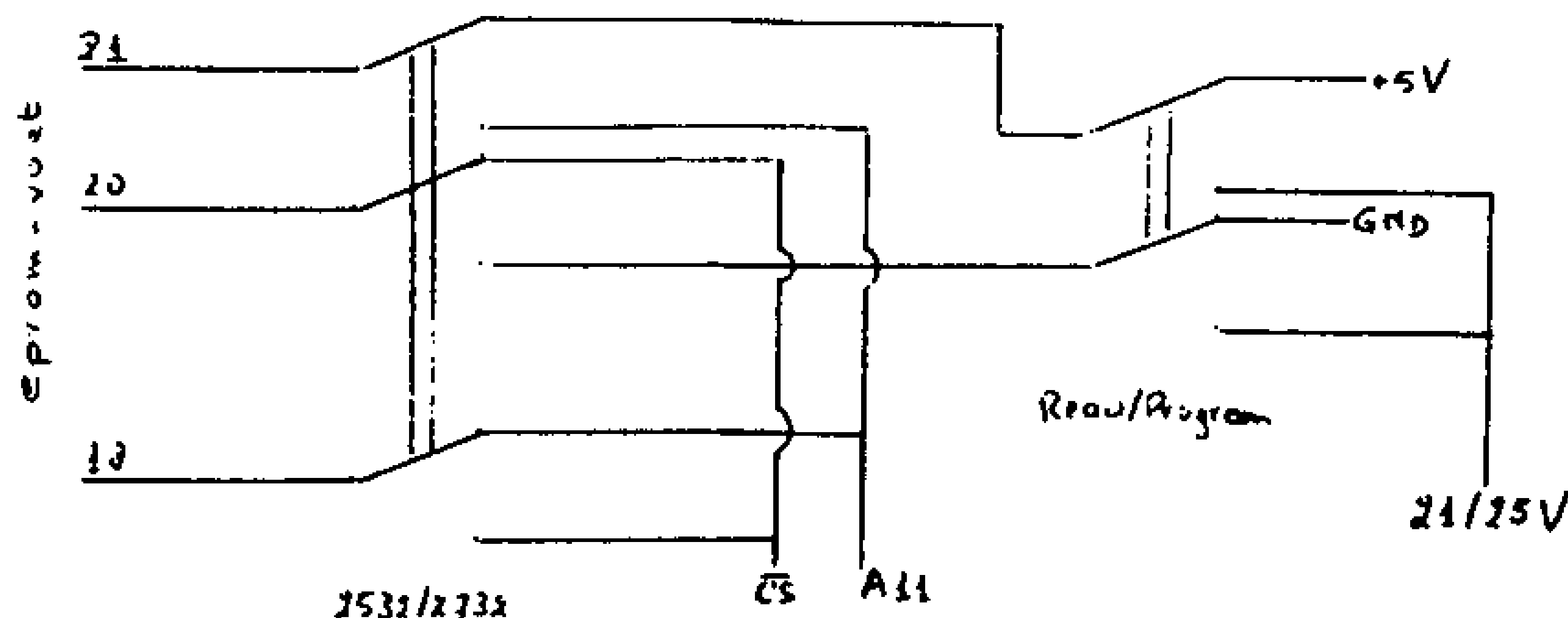
Voor de graphicskaart is een monitor noodzakelijk wil de graphics goed zictbaar zijn.
Voor de tekstkaart is een videomonitor gewenst maar met een goede TX televisie is voor tekst een heel aardig resultaat te bereiken.

## *** EIGEN ONTWIKKELINGEN ***

Een punt wat hier helemaal los van staat is het volgende: Wanneer u zelf een hardware aanpassing aan de computer heeft gedaan (dit hoeft in het geheel niet revolutionair te zijn), of een idee hebt, schroom dan niet dit ontwerpje, met een korte beschrijving en schema! IN TE STUREN (redaktieadres). Zeer waarschijnlijk hebben meer mensen er wat aan!!

Een eenvoudige doorverbinding kan soms al een hele boel problemen oplossen!! Het idee van de te ontwikkelen 80 koloms kaart is ook zo ontstaan

Het is inderdaad mogelijk om met eenvoudige schakelaars de programmer zodanig om te bouwen dat je met één en hetzelfde programma zowel de 2532- als 2732-eprom kunt programmeren. Daartoe moet je een drievoudige (of een tweevoudige en een enkelvoudige of drie enkelvoudige) en een tweevoudige om-schakelaar monteren. Deze verbind je met de rest van de programmer als volgt:




De 2532/2732-schakelaar staat in 2532-mode getekend en de read/program-schakelaar in de read-mode. Let er overigens op dat er "gewone" eproms bestaan en zgn. A-typen. Deze laatste vragen 'n programmeerspanning van 21 Volt en gaan bijna zeker kapot bij 25 Volt. Andersom schijnt het zo te zijn dat je de gewone typen eveneens op 21 Volt kunt programmeren.
Verder is het niet algemeen bekend dat de jumper voor 2716's op de kaart niet voldoende is om deze eproms te programmeren. Aangezien de timings en pulsen anders zijn, zal ook het programma aangepast moeten worden.

City bomber is een spelletje, dat gebruik maakt van de elders in dit nummer van Acorntjesbrood uitvoerig besproken GAGS-2 toolkit van Gerrit Hillebrand. Het is voornamelijk geschreven om eens te kijken wat GAGS voor mogelijkheden biedt om een spelletje te maken. Het blijkt dat het schrijven van een spelletje nu veel en veel eenvoudiger wordt. Gerrit is dus uitstekend in zijn opzet geslaagd. Iedereen kan nu in Basic, dus zonder machinetaal routines, heel aardige spelletjes schrijven die vroeger nauwelijks realiseerbaar waren.

## Beschrijving City Bomber:

Je vliegt in een vliegtuig over een vijandige stad met als speciale opdracht om deze stad te bombarderen. Met de SHIFT-toets kun je twee bommen op de stad gooien. Als er een flatgebouw wordt getroffen, zal dat een aantal etages instorten. Elke etage levert 1 punt op.
Door de in de vijandige stad gelegerde troepenmacht word je bestookt met afweergeschut. Als je hierdoor wordt geraakt, stort je neer met je vliegtuig. Je hebt in totaal drie vliegtuigen (='levens').
Rechtsboven zie je hoeveel je er nog over hebt (in het begin dus twee). Als je neerstort, wordt één van de resterende vliegtuigen rechtsboven startklaar gemaakt.
Om het afweergeschut te ontwijken, kun je van de 'Turbo-Boost' gebruik maken door op de CTRL-toets te drukken. Je zult dan kortstondig een stuk sneller vliegen en kunt zo dus (misschien) aan de dood ontsnappen. Het gebruik van de Turbo-Boost kost je per keer wèl 10 strafpunten.
Door brandstofgebrek gaat je vliegtuig steeds lager vliegen. Pas dus op dat je niet tegen een flatgebouw vliegt! Je stort dan neer. Zorg er dus voor dat je de hoogste flatgebouwen als eerste in puin gooit.
Als je de hele stad hebt vernietigd, word je door de automatische piloot veilig aan de grond gezet. Daarna moet je een grotere (=hogere) stad bombarderen. Als je met één vliegtuig de hele stad hebt platgegooid, krijg je een speciale bonus van 50 punten!
Boven in beeld zie je de huidige score en de hoogste tot nu toe behaalde score.



```
 10 REM CITY BOMBER
 20 REM GAGS-2 NOODZAKELIJK
 30 REM AUTEUR: RONALD BOERS
 40 GOSUB k
 50 P."YOU ARE FLYING OVER A CITY,WHICH"
 60 P."HAS TO BE DESTROYED BEFORE YOU"'
 70 P."LAND WITH YOUR PLANE."'
 80 P."THERE WILL BE SOME ANTI-AIRCRAFT"
 90 P."MISSILES LAUNCHED AT YOU."''
100 P."USE (SHIFT) TO DROP YOUR BOMBS."'
110 P."USE (CTRL) TO START TURBO BOOST"'
120 P."TO AVOID THE MISSILES.THIS WILL"'
130 P."COST YOU 10 POINTS."'
140 REM SPRITE DEFINITIONS
150 REM STORAGE AREA:#4800-#5000
160 BASE#48
170 CREATE PLANE,#0C,#07,#03,#7F,#FF,#E3,#C7,#8C,0,0,#C0,#FF,#FE,#C0,0,0
180 CREATE BOMB,#60,#7E,#60,0,0,0,0,0,0,0,0,0,#C0,#FC,#C0,0
190 CREATE MISSILE,#28,#10,#10,#10,#10,#10,0,0,0,0,#28,#10,#10,#10,#10,#10
200 CREATE EXPLOSIE,#03,#04,#09,#03,#12,#05,#08,#02,#20,#84,#58,#50,#28,#30,#90,#20
210 CREATE ROCKS,#FF,#9B,#E5,#9A,#66,#39,0,0,#FE,#9A,#34,#CE,#34,#68,#C0,0
220 CREATE ZERO,#38,#C6,#C6,#C6,#C6,#C6,#C6,#38,0,0,0,0,0,0,0,0/A:1
230 CREATE ONE,#3C,#18,#18,#18,#18,#38,#18,#08,0,0,0,0,0,0,0,0/A:2
240 CREATE TWO,#FE,#62,#30,#18,#0C,#86,#C6,#7C,0,0,0,0,0,0,0,0/A:3
250 CREATE THREE,#7C,#C6,#6C,#18,#38,#18,#C,#FE,0,0,0,0,0,0,0,0/A:4
260 CREATE FOUR,#06,#06,#86,#FF,#C6,#66,#36,#1E,0,0,0,0,0,0,0,0/A:5
270 CREATE FIVE,#7C,#C6,#86,#06,#FC,#C0,#C0,#FE,0,0,0,0,0,0,0,0/A:6
280 CREATE SIX,#7E,#C3,#C3,#C3,#FE,#C0,#C0,#7E,0,0,0,0,0,0,0,0/A:7
290 CREATE SEVEN,#18,#18,#18,#18,#0C,#06,#83,#FF,0,0,0,0,0,0,0,0/A:8
300 CREATE EIGHT,#7E,#C3,#C3,#C3,#7E,#C3,#C3,#7E,0,0,0,0,0,0,0,0/A:9
310 CREATE NINE,#7E,#C3,#83,#03,#7F,#C3,#C3,#7E,0,0,0,0,0,0,0,0/A:10
320 CREATE FLAT1,#FF,#AB,#FF,#AB,#AB,#FF,#AB,#FF,#FE,#2A,#FE,#2A,#2A,#FE,#2A,#FE/A:11
330 CREATE FLAT2,#FF,#AD,#FF,#AD,#FF,#AD,#AD,#FF,#FE,#B6,#FE,#B6,#FE,#B6,#B6,#FE/A:12
340 CREATE FLATTOP,#7F,#1C,#08,#08,#08,#08,0,0,#FC,#90,#90,#F0,0,0,0,0
350 CREATE HI,#C3,#C3,#C3,#C3,#FF,#C3,#C3,#C3,#7E,#18,#18,#18,#18,#18,#18,#7E
360 CREATE SC,#7E,#C3,#83,#3E,#7C,#C1,#C3,#7E,#3E,#63,#61,#60,#60,#61,#63,#3E
370 CREATE OR,#7C,#C6,#C6,#C6,#C6,#C6,#C6,#7C,#C3,#C6,#CC,#D8,#FE,#C3,#C3,#FE
380 CREATE E,#FF,#C1,#C0,#C0,#FC,#C0,#C1,#FF,0,#18,#18,0,0,#18,#18,0
390 CREATE/P:1 #55,#AA,#55,#AA,#55,#AA,#55,#AA
400
410 P.'"PRESS SPACE BAR TO START."
420 LINK#FFE3
430 DIM HH(20);REM HEIGHT BUILDINGS
440 P=#B000;@=0
450 Q=0;REM HIGH-SCORE
460 U=2;REM NUMBER OF SPARE PLANES
470 C=0;S=0;REM SCORE
480 H=8;REM MAX HEIGHT BUILDINGS
490 FOR I=0 TO 20;HH(I)=0;NEXT
500 X=0;A=1;L=2;D=1
510 Z=0;B=0;V=0;Y=0
520
530 REM DRAW CITY
540 CLEAR4
550 CUBE 1,0,0,0,255,8
560 PAINT 1,1,1
570 FOR I=32 TO 220 STEP 16
580  K=ABSRND%5;IF K>2;K=K-2
590  T=3+ABSRND%(H-3)
```

```
 600   L=L+1
 610   IF K=0;GOTO s
 620   HH(L)=T+1
 630   Z=Z+HH(L)
 640   FOR J=1 TO T
 650    IMAGE (K+10),I,(8+J*8)
 660   NEXT J
 670   IMAGE:FLATTOP,I,(8+J*8)
 680sNEXT I
 690 BLOCK 1,3,170,250,16
 700 CUBE 1,0,1,168,253,19
 710 IMAGE:SC, 7,181
 720 IMAGE:OR,24,181
 730 IMAGE:E ,41,181
 740 N=S;W=57;GOSUB r
 750 IMAGE:HI,103,181
 760 IMAGE:SC,119,181
 770 IMAGE:OR,136,181
 780 IMAGE:E ,153,181
 790 N=Q;W=169;GOSUB r
 800 IF U>0;IMAGE:PLANE,213,181
 810 IF U>1;IMAGE:PLANE,232,181
 820 SET:PLANE,0,160
 830
 840 REM MAIN LOOP
 850 DO
 860 IF Y=0;IF A;IF P?1&#40=0;Y=4;GOSUB e;SOUND 200,8
 870 SHOVE:PLANE,8,0
 880 IF Y;SHOVE:PLANE,8,0;Y=Y-1
 890 IF Y=0;A=(P?1&#40<>0)
 900 IF V=0;IF ABSRND%5=1;V=1;SET:MISSILE,(ABSRND%240),16
 910 IF V;SHOVE:MISSILE,0,8
 920 POS:MISSILE,E,F
 930 IF F>160;UNSET:MISSILE;V=0
 940 ATHIT:PLANE,:MISSILE;SET:EXPLOSIE,E,F;V=0;O=1;GOTO c
 950 POS:PLANE,E,F
 960 PIXEL (E+16),(F-4),O
 970 IF O=0;GOTO m
 980 POS:PLANE,I,J
 990 UNSET:PLANE
1000 E=16*((E+16)/16)
1010 BLOCK 0,E,(F-7),16,8
1020 SET:PLANE,I,J
1030 IMAGE:ROCKS,E,F
1040 V=0;GOTO c
1050mIF B=0;IF P?1&#80=0;B=1;SET:BOMB,(16*(E/16)),(F-8)
1060 IF E>240;SHOVE:PLANE,8,-8
1070 IF B=0;PAUSE 2;GOTO c
1080 POS:BOMB,E,F
1090 T=HH(3+(E-32)/16)
1100 IF F>8*(T+1);SHOVE:BOMB,0,-8;GOTO c
1110 REM BOMB HIT SOMETHING
1120 B=0
1130 UNSET:BOMB
1140 SET:EXPLOSIE,E,F
1150 FOR I=20 TO 100 STEP 20
1160  MODE 7
1170  SOUND I,3
1180  MODE 8
```

```
1190  SOUND (101-I),3
1200 NEXT I
1210 UNSET:EXPLOSIE
1220 IF T=0;GOTO c
1230 POS:MISSILE,G,M
1240 IF M<=8*(T+1);IF ABS(G-E)<=16;UNSET:MISSILE;X=1
1250 R=1+ABSRND%(T-T/3)
1260 FOR I=1 TO R
1270  BLOCK 0,E,(F-7),16,8
1280  SOUND (I*10),6
1290  F=F-8
1300 NEXT I
1310 HH(3+(E-32)/16)=T-R
1320 Z=Z-R;S=S+R
1330 IF T<=R;GOTO d
1340 BLOCK 0,E,(F-7),16,8
1350 IMAGE:ROCKS,E,F
1360dIF X;SET:MISSILE,G,M;X=0
1370 N=S;W=57;GOSUB u
1380cUNTIL Z=0 OR O
1390 IF O;GOTO h
1400 UNSET:MISSILE
1410 FOR J=1 TO 10
1420  FOR I=10 TO 100 STEP 5
1430   SOUND (10+ABSRND%I),(15-J)
1440  NEXT I
1450 NEXT J
1460 DO
1470  SHOVE:PLANE,8,-1
1480  PAUSE 4
1490  POS:PLANE,E,F
1500 UNTIL F<18
1510 FOR I=80 TO 10 STEP -1
1520  PAUSE 4
1530  SHOVE:PLANE,(I/10),0
1540  POS:PLANE,E,F
1550  IF E>240;CARRY:PLANE,1,17
1560 NEXT I
1570 PAUSE 100
1580 GOSUB k
1590 P."WELL DONE!!"'"MISSION COMPLETED!!"''
1600 IF D;P."SPECIAL BONUS: 50 POINTS."'';S=S+50
1610 P."LET'S TRY A LARGER CITY..."'
1620 PAUSE 400
1630 H=H+1;GOTO 490
1640hREM PLANE HIT SOMETHING
1650 UNSET:BOMB
1660 UNSET:MISSILE
1670 FOR I=1 TO 10
1680  FOR J=9 TO 60 STEP 4
1690   MODE 7
1700   SOUND J,10
1710   MODE 8
1720   SOUND(69-J),10
1730  NEXT J
1740 NEXT I
1750 UNSET:EXPLOSIE
1760 B=0
```

```
1770 DO
1780  SHOVE:PLANE,8,-8
1790  SOUND (200-F),4
1800  PAUSE 4
1810  POS:PLANE,E,F
1820 UNTIL F<16
1830 IMAGE:EXPLOSIE,E,F
1840 FOR I=10 TO 200 STEP 10
1850  SOUND I,(I/10)
1860  SOUND (210-I),(I/10)
1870 NEXT I
1880 PAUSE 100
1890 U=U-1;IF U<0;GOTO 2080
1900 I=213+U*19
1910 BLOCK 1,I,174,16,8
1920 SET:PLANE,I,181
1930 DO
1940  SHOVE:PLANE,0,-1
1950  SOUND (F/2),8
1960  PAUSE 3
1970  POS:PLANE,E,F
1980 UNTIL F=160
1990 FOR I=10 TO 82
2000  SHOVE:PLANE,(I/10),0
2010  SOUND (100-I),(15-I/10)
2020  PAUSE 3
2030 NEXT I
2040 POS:PLANE,E,F
2050 SHOVE:PLANE,(8-E%8),0
2060 D=0
2070 GOTO 840
2080 GOSUB k
2090 P."                game"$#80"over"$#8E''
2100 P."YOU SCORED "S" POINTS."''
2110 IF S>Q;P."YOU HAVE THE HIGHEST SCORE!!"'';Q=S
2120 P."HIGH-SCORE: "Q''
2130 P."press"$#80"space"$#80"bar"$#80"to"$#80"start"$#8E''
2140 LINK#FFE3;GOTO 460
2150eIF S<10;S=0
2160 IF S>=10;S=S-10
2170 N=S;W=57
2180uREM UPDATE SCORE
2190 T=N;N=C;GOSUB r
2200 N=T
2210rI=(N%10000)/1000+1
2220 J=(N%1000)/100+1
2230 K=(N%100)/10+1
2240 L=N%10+1
2250 IMAGE I,W,181
2260 IMAGE J,(W+10),181
2270 IMAGE K,(W+20),181
2280 IMAGE L,(W+30),181
2290 C=S
2300 RETURN
2310kP.$12;?#E1=0
2320 P."*-*-*-*-*-*-*-*-*-*-*-*-*-*-*"'
2330 P."**-*-*-*- CITY BOMBER -*-*-*-**"'
2340 P."*-*-*-*-*-*-*-*-*-*-*-*-*-*-*"''
2350 RETURN
```

Alle goede bedoelingen ten spijt, kan het gebeuren, dat je bij gebruik van DBASE V2.0 per ongeluk het Quit-commando uitvoerd of op Break drukt.
De datafile in het geheugen wordt dan bij opnieuw starten met LINK #8DD9 niet meer als zodanig herkend.

Hier volgt een truc om dit probleem op te lossen:
De opdracht LINK #8DD9 verzorgt de start van DBASE V2.0 en verwijst naar een subroutine die drie dingen doet:

-op #2800 komt #00

-op #202 komt #C5

-op #203 komt #8F

De #00 op #2800 zorgt voor het probleem dat de datafile niet meer gelezen kan worden.

Bij aanmaak van een nieuwe datafile m.b.v. Create en Insert wordt de #00 op #2800 vervangen door een ander getal, zodat een datafile nu herkend wordt. Na een Break of een Quit gaat dit andere getal niet verloren.

Je gaat pas de boot in bij het opnieuw starten m.b.v. LINK #8DD9.

Overigens wordt na een Break de inhoud van #202 vervangen door #D8 en #203 door #C9.

```
?#202=#C5:?#203=#8F:LINK #8DDC
```

en alles is weer hersteld (op #8DDC staat de verwijzing naar de eerstvolgende subroutine).

Noot van de redaktie:
Het betreft hier het database programma welke is gepubliceerd in ACORN NIEUWS nr. 6 van 1983. Het in dit nummer gepubliceerde programma is een nieuwere versie. Het is de vraag of bovenstaande voor deze ook geldig is.

# HET *OPT STATEMENT UIT DE TELEC DOS ROM

In zowel de TELEC als de ACORN DOS ROM was blijkbaar nog voldoende ruimte over om er een extra * commando in te stoppen. In de ACORN DOS zit er *VDU, welk door een aantal mensen gebruikt is als voorbeeld voor de software voor de 80 koloms VDU kaart.
In de TELEC DOS ROM zit in de vrije ruimte het *OPT commando en een soort bootstrap loader voor autostart DOS.
Eerst het *OPT commando. Dit commando kan gevolgd worden door één van de cijfers 0,1,2,3. ( Van andere cijfers wordt net zolang 4 afgetrokken tot de rest tussen 0 en 3 ligt!) Dit getal wordt zowel op de in de drive zittende floppy geschreven als in het high nibble van #2106. Dit is alles wat het *OPT commando doet, simpel commando'tje dus.

Maar nu de truc!

Als de ATOM na een break (en na *DOS) langs #EFF3 gestuurd wordt via *GO #EFF3 of LINK #EFF3 kunnen er 4 dingen gebeuren , afhankelijk van de optie van de schijf , nadat eerst het beeld schoon geveegd is en de tekst ACORN ATOM DOS op beeld verschijnt.

Optie = 0: Er gebeurd verder niets behalve dat de cursor nu * is en de ATOM alleen *commando's accepteerd. (Terug naar normale Basic via GO C558)

Optie = 1: Het programma met de naam BOOT wordt van de schijf geladen en er wordt verder gegaan als bij optie = 0.(Vgl. *LOAD BOOT)

Optie = 2: De file met de naam BOOT wordt van schijf geladen en als dit een machinetaal programma of een autostart basic programma is, dan wordt het ook meteen uitgevoerd.(Vgl. *BOOT)

Optie = 3: De file met de naam BOOT wordt uitgevoerd als een ASCII file. Opm. Alleen * commando's zonder * in deze ASCII file. Te omzeilen door GO C558 te geven, waarna weer alle 'normale' invoer mogelijk is.(Vgl. *EXEC BOOT)

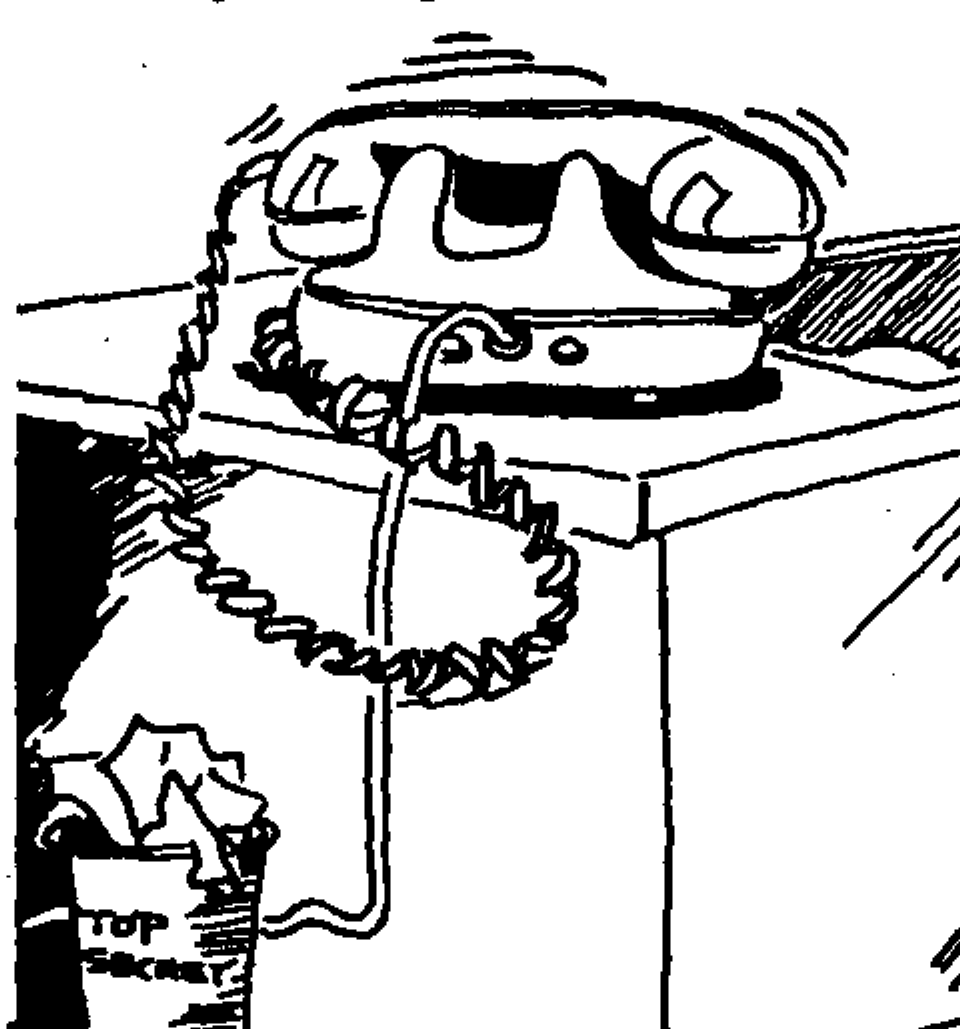
Voorbeelden van gebruik zijn: maak een schijf met uitsluitend tekstfiles, zet hier de Wordpack op onder de naam BOOT en geef de schijf de optie 2. Maak een ASCII file met daarin allerlei settings van vectoren zoals die voor deze specifieke ATOM gelden. Save deze file onder de naam BOOT en geef de schijf de optie 3.

De optie van de schijf verschijnt altijd rechts boven in beeld als er een *CAT van de schijf gevraagd wordt.

Bovenstaande zou gebruikt kunnen worden om een soort turn-key ATOM te maken. Hiervoor moet dan de monitor ROM van de ATOM veranderd worden zodat na een break er automatisch naar #EFF3 gesprongen wordt en de ATOM zichzelf opstart met het programma BOOT wat op de schijf staat die op dat moment in de drive zit. Nog niet opgelost is het probleem wat er moet gebeuren als er helemaal geen diskdrive aanhangt of er geen floppy in zit. In dat geval blijft de ATOM tot in lengte van dagen hangen, al dan niet met draaiende drive motor. De enige redding is dan break, maar dan begint het feest weer van voren af aan.
Daarom geef ik de voorkeur aan de standaard ATOM monitor ROM en met behulp van het extra statement BOOT (dank aan P-CHARME) worden dan de *OPT mogelijkheden benut.

*Dit overzicht van de derde jaargang Acorn Nieuws is gesorteerd naar onderwerp. De letters in de tweede kolom betekenen het volgende:*

*A = ASSEMBLER*
*B = BASIC*
*F = FORTH*
*S = SCHEMA*



DRIN!
TOP SECRET